Summary

This project concerns the area of computerised meetings management and how a computerised system can benefit committee members.

The primary aim of the project was to produce a generic prototype meetings management system that would make the organisation and distribution of documents associated with meetings more effiecient. This involved looking at existing systems, interviewing possible beneficiaries and modelling committee procedure. Due to the nature of the project, numerous ideas came to light in terms of future improvements and these have been discussed.

Acknowledgements

Initially let me state my thanks to Stuart Roberts for his continuing support throughout this project and for making himself available at short notice to discuss any issues that had cropped up.

Thanks must also be given to John Davy and James Craig for their time within the interviews and the information, suggestions and feedback they gave.

I would also like to thank Tony Cohn for providing useful suggestions (and criticism) within the progress meeting that helped me implement some advanced functionalities later in the development.

I like to thank my parents for helping me through University and for proof reading the final report.

Above all I would like to thank Claire, for putting up with me through what must have been the most stressful time for her too.

Thanks to all my friends for helping to test the system and to anyone who helped proof read the final project report or test the prototype system.

Acronyms

AM	Agile Modelling
API	Application Programming Interface
ASP	Active Server Pages
BCNF	Boyce-Codd Normal Form
CGI	Common Gateway Interface
CPU	Central Processing Unit
CW	Coursework
DB	Database
DBMS	Database Management System
EOC	Equal Opportunities Committee
ER	Entity Relation
FK	Foreign Key
GNU	GNU's Not Unix (recursive acronym)
GPL	General Public License
GUI	Graphical User Interface
HCI	Human Computer Interaction
HTML	Hypertext Mark-up Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
IDE	Integrated Development Environment
IE	Internet Explorer
IIS	Internet Information Services
IS	Indexing Service
JDBC	Java Database Connectivity
JSP	Java Server Pages
JVM	Java Virtual Machine
MM	Meetings Management
MS	Microsoft
OCI	Oracle Call Interface
ODBC	Open Database Connectivity
OOD	Object Orientated Development
PC	Personal Computer
PHP	Hypertext Preprocessor
PK	Primary Key
RAD	Rapid Application Development

RUP	Rational Unified Process
SE	Software Engineering
SLTC	School Learning and Teaching Committee (UoL)
SoC	School of Computing
SQL	Structured Query Language
UML	Unified Modelling Language
UoD	Universe Of Discourse
UoL	University of Leeds
URL	Uniform Resource Locator
XML	Extensible Markup Language
XSL	XML Stylesheet Language

Contents

1. Introduction 1

1.1 Project Overview 1
1.2 The Need 1
1.2.1 Motivation 1
1.2.2 Meetings 1
1.2.3 The Associated Documentation 1
1.3 Aim and Requirements 2
1.3.1 The Project Aim 2
1.3.2 Minimum Requirements 2
1.4 The Deliverables 3
1.5 The Schedule 3
1.6 Structure Of The Report 4

2. Requirements Gathering and Analysis 6

2.1 A Set of User Requirements 6
2.1.1 Introduction 6
2.1.2 Target users 6
2.1.3 Information Gathering Techniques 7
2.1.3.1 A Review of Proven Techniques 7
2.1.3.2 SQIRO 7
2.1.3.3 Conclusion 8
2.1.3.4 Interview Techniques and Structure 8
2.1.4 User Requirements 9
2.1.4.1 User Requirements Obtained Through Interviews 9
2.1.4.2 User Requirements Obtained Via Other Means 11
2.2 System Requirements 11
2.2.1 Introduction 11
2.2.2 A Standalone system 12
2.2.3 A True Client/Server System 12
2.2.4 A Web-Based System 12
2.2.5 Conclusion 13
2.3 Storage Requirements 13

3. Design and Methodology 15

3.1 Possible Methodologies 15

3.1.1 Introduction 15
3.1.2 The Waterfall Model 15
3.1.3 The Spiral Model 16
3.1.4 The Incremental Model 16
3.1.5 RAD 16
3.1.6 Agile Techniques 16
3.1.7 RUP 17
3.1.8 Conclusion 17
3.2 Database Design 18
3.2.1 ER-Diagram 18
3.2.2 Design Assumptions 19
3.2.3 Relations 20
3.2.4 Entities 20
3.2.5 Normalisation 22
3.2.6 Integrity 22
3.2.7 Index Optimisation - A Compromise 22
3.3 Program Design 23
3.3.1 Front End 23
3.3.1.1 Introduction 23
3.3.1.2 HCI issues 23
3.3.1.3 The GUI 24
3.3.2 Back End 24
3.3.2.1 Database Connectivity 24
3.3.2.2 ODBC and DB Drivers 25
3.3.2.3 Uploading Documents 25
3.3.3 Design For Re-Use 25
3.3.4 Possible Security Models. 25

4. Background Research 27

4.1 Computerised Meetings Management 27
4.1.1 System Necessities 27
4.2 Existing Systems 28
4.2.1 Introduction 28
4.2.2 The Current Trend 28
4.2.3 Evaluation criteria 28
4.2.4 Conclusion 29
4.3 Possible Technologies 30

4.3.1 Introduction 30
4.3.2 An Abundance of Possible Languages 30
4.3.2.1 Introduction to Server Side Technologies 30
4.3.2.2 ASP 30
4.3.2.3 PHP 30
4.3.2.4 CGI 31
4.3.2.5 JSP/Servlets 31
4.3.2.6 Evaluation and Conclusion 31
4.3.3 Choice of DBMS 31
4.3.3.1 MySQL 32
4.3.3.2 PostgreSQL 32
4.3.3.3 SQL Server 2000 32
4.3.3.4 Evaluation and Conclusion 32
4.3.4 Client Side Technologies 32
4.3.4.1 The Need for Client Side Validation 32
4.3.4.2 Applets 33
4.3.4.3 VBScript 33
4.3.4.4 JavaScript 33
4.3.5 The Suitability of Active Rules 34
4.3.6 Development Tools and Environments 34
4.3.6.1 SE Environment 34
4.3.6.2 DB Environment 35

5. Implementation 36

5.1 Development Environment 36
5.2 Good Programming Practice 36
5.3 Summary of Development 37
5.3.1 Page Implementation 37
5.3.2 Uploading Documents to the Server 38
5.3.3 Security Model 38
5.4 Problems Encountered 38
5.4.1 Connection Issues 38
5.4.2 Tomcat Configuration 39
5.4.3 Issues Relating to Design 39

6. Evaluation 41

6.1 Testing of the MM System 41

6.1.1 Input Validation 41
6.1.2 Performance 42
6.1.3 Document Upload 42
6.1.4 Security 42
6.2 Evaluation of the MM System 43
6.2.1 Essential User Requirements 43
6.2.2 Desirable User Requirements and Improvements 45
6.2.3 The End System 45
6.2.3.1 Overview 45
6.2.3.2 Comparison to Existing Systems 46
6.2.3.3 Security Model 47
6.2.3.4 Potential User Feedback 47
6.2.3.5 Expandability 47
6.2.4 Conclusion 48
6.3 Evaluation of the Project Schedule/Plan 48

7. Conclusion 50

7.1 Future Developments 50
7.2 Project Conclusion 52

Bibliography 53

Appendix A – Reflection 55

Appendix B – Schedule 57

B.1 Original Schedule 57
B.2 Modified Schedule 59

Appendix C – Interview Transcripts 60

C.1 James Craig 60
C.2 John Davy 63
C.3 Stuart Roberts 67

Appendix D – The System's Main Page 71

Appendix E – DB Diagram 72

Appendix F – User Manual 73

Chapter 1

Introduction

1.1 Project Overview

This project is concerned with Computerised Meetings Management, that is, it is related to improving the document management associated with organising meetings. The primary aim of this project is to produce a prototype version of a generic Meetings Management (MM) System; hence the problem to be solved is that of improving the organisation and distribution of documents to make meetings more efficient**.** As MM is a large area and improved MM helps benefit numerous organisations, numerous systems exist. A few of these systems are reviewed in section 4.2.

1.2 The Need

1.2.1 Motivation

The motivation for this project appears fairly obvious at first. Most employment involves meetings of some type, whether formal or informal. The need for improved organisation and more efficient document management obviously increases with large companies. As meetings are such an important part of professional careers and academia then the quest to make the administration of them more efficient and less prone to errors is surely a valid one. The production of a system that can automate simple monotonous tasks associated with MM can only be beneficial and will allow the people usually tied up with such work to contribute more creatively to the actual meeting itself, while helping to lighten the load by acting as an aid memoir for participants to remember to carry out the tasks attributed to them.

1.2.2 Meetings

The word ‘meeting’ is such a general one that a more precise explanation is needed. A suitable definition in terms of this project, is, “An organised assembly for transaction of business” [1]. This project is concerned with organised meetings and committee procedure and mainly with the distribution of related documents that inevitably proceed and follow a meeting.

1.2.3 The Associated Documentation

The process of organising a meeting and indeed that of holding the meeting itself generates a multitude of documents. These documents may include minutes, agendas, and other supporting papers. The person usually responsible for producing these documents is the committee secretary.

Meetings occur on a regular basis for most committees and the associated documentation often shares numerous similarities. This can lead to the production of these documents becoming increasingly repetitive.

1.3 Aim and Requirements

1.3.1 The Project Aim

The primary aim of this project is to produce a generic MM system. This system will help to automate some of the repetitive and time consuming tasks associated with MM, including the distribution of minutes and other documents associated with meetings. User requirements will be gathered from selected staff within the School Of Computing (SoC) at the University of Leeds. The SoC will be used as a case study to evaluate the software solution against. The main objectives of the project are listed below; completion of the project aim will be gained by meeting these objectives.

1. To conduct a review of existing MM systems and evaluate them against the user requirements expressed from some selected staff of the SoC that are associated with MM.
2. To gather requirements from selected staff within the SoC, who are associated with any administrative task associated with organising meetings.
3. To evaluate appropriate tools to implement a MM system including the applicability of active rules for these systems.
4. To produce a generic software system for MM using proven methodologies and technologies.
5. Produce a system that allows for management of tasks or actions.
6. To evaluate the software system against the requirements expressed from objective 2, above.

The minimum requirements will be met upon completion of the 4th objective outlined above. The remaining objectives would help to provide a system that can be further enhanced to incorporate extra features and provide advanced functionality to the end user.

1.3.2 Minimum Requirements

A more concise list if the minimum requirements can be found below, these are taken from the main objectives outlined in the previous section.

1. A review of existing meeting management systems.
2. Gather a set of requirements from end users within the SoC.
3. Produce a prototype of a generic MM system.
4. Evaluate the solution in an accurate and systematic way.

Point three is really a set of minimum requirements in its own right. The functionality that is needed in order to show minimum requirement three has been reached is explained in section 2.2.1.

1.4 The Deliverables

The project has three main deliverables, these are:

1. A working prototype MM system that meets the stated minimum requirements.
2. A user manual for the MM system.
3. A project report which fully documents the project.

1.5 The Schedule

The project needed a formal schedule devising so that it can be approached in a systematic way and milestones could be set out and recorded. The initial schedule that was presented in the mid-project report is also shown below. Alterations made to the schedule are documented and fully reviewed in section 6.3. The milestones are shown as **Num1** for milestone 1 etc.

1. Submit Preference form (**Num1**). Project decided - Can now start research.
2. Discussion and alteration of initial project idea with supervisor.
3. Begin researching existing MM technologies, different tools/technologies available and their relevance. Primarily concentrate on information capture techniques in the early stages. Later stages of research will be orientated around the chosen methods used to design and produce the software artefact.
4. Complete Aim and Minimum requirements form (**Num2**). - Completed.
5. Start to gather user requirements in the most appropriate way.
6. Model these requirements and get feedback on them to ensure that they are correct.
7. Complete Mid-Project report (**Num3**). – Complete
8. Review possible methodologies that could be used and decide on the most appropriate one.
9. Complete the review of existing MM systems and their relevance to the user's requirements from within the SoC (**Num4**) - A further Minimum requirement is completed.
10. Begin the production and implementation of the generic MM system.
11. Submit draft chapter and table of contents. (**Num5**) – Chance for feedback from supervisor.
12. Conduct progress meeting with assessor and supervisor. (**Num6**) – Chance for feedback from the report and the actual system.
13. Complete the MM system and evaluation. (**Num7**) - Minimum requirements have now been met.

14. Extend the existing system to incorporate management of actions or tasks and evaluate it. (**Num8**) – Minimum requirements exceeded.
15. Complete write up of final report.
16. Submit Final report. Proof reading. (**Num9**) – Project Complete.

A Gantt chart graphically showing the original schedule outlined above, and the changes made to the actual project schedule can be found in Appendix B.

1.6 Structure Of The Report

This chapter and the subsequent chapters contained within the project report are structured as follows:

This Chapter introduced the primary aims and objectives of the project and provides brief details concerning the subject area of MM. The problem is stated and justification is provided as to why it is worthy of investigation. The planned schedule was then presented outlining milestones within the project. Finally this section of the chapter was given, which outlines the structure and content contained within the rest of the project report.

Chapter two is concerned with gathering a set of user requirements from possible users within the SoC. This includes background research into suitable information gathering techniques to be used for this purpose. The user requirements are then stated and some key design decisions are made as to the most viable structure for the final software artefact to adhere to. A brief explanation is then given about the storage requirements of the system.

Chapter three focuses on the design of the system. A suitable methodology is decided upon for the remainder of the project. This attempts to provide full justification, before the design of the underlying database (DB) is shown along with stated relations, attributes and the ER-Model from which the design grew. The Program design is then explained and issues relating to HCI and DB connectivity are discussed.

Chapter four focuses on background information relating to MM. Initially this involves a review of selected systems. This is followed by research into the possible technologies that could be used for the production of the end system; including scripting/programming languages and choosing a suitable DBMS. The suitability of active rules is then discussed before the choice of development tools and environments are stated and justified.

Chapter five discusses the development of the system and underlying database. This includes the main issues that arose during the implementation stage of the project.

Chapter six deals with testing and evaluating the solution produced. The purpose of the testing is to make sure that the system performs as expected. Then the system is evaluated against the original user requirements and the system produced is compared to the existing systems available. A detailed discussion is then presented on the security issues of the system and any perceivable flaws.

Chapter seven then concludes the whole project. This section also includes an extensive list of possible areas for future work/enhancements, related to the project, and suggests possible approaches that could be used to implement the enhancements.

Chapter 2

Requirements Gathering and Analysis

This chapter aims to explain how the user requirements were gathered and why the chosen information gathering techniques were employed.

2.1 A Set of User Requirements

2.1.1 Introduction

As the project involves the production of a generic prototype MM System, a set of user requirements was needed. These would be used to form a specification, on which to build the design, which would then lead to the implementation stage.

2.1.2 Target Users

Before user requirements can be formulated, target users need to be identified. These users need to be people who are experienced within this field and play a dominant role with the administrative tasks of organising and running meetings. These 'target users' are also people that may reap the benefit of using such a system. However it is also worth re-iterating that the produced system would not necessarily follow their perceived requirements exactly as there is no distinct problem owner, but more that the requirements will be used as template for the prototype to adhere to. The reason for this is that as stated in the last chapter the system is not being specifically designed for a particular set of users, but is being designed as a generic system that could be used by anyone involved with the administration of meetings. The system would also benefit those members that attend meetings by providing a central place for them to access the associated documentation. So the target users of such a system can be though of as any member of a committee that is involved with meetings with those members being involved with the administrative tasks of greater benefit.

A set of requirements was still needed and the following possible users were identified within the SoC:

Mr. James Craig (JC)	Head of Clerical and Secretarial Support within the SoC. Takes minutes for the Staff meetings and the SMB (School Management Board).
Dr. John Davy (JD)	Chairs the SLTC (School Learning and Teaching Committee) checks the minutes and produces the agendas.

Dr. Stuart Roberts (SR) Produces agendas and minutes for the Information Systems Database (IS-DB) Course Team and agendas for the Equal Opportunities Committee. Previously chair of the Software Engineering (SE) subgroup of the SLTC, which is now disbanded.

These potential 'target users' were perceived to be of differing backgrounds and hence would be more likely to offer a wider range of information. For example one of JD's primary roles is lecturing and research, whereas JC is primarily concerned with administrative duties and his role within the school is that of Senior Administrator Officer. SR was also considered as a good candidate for possible user requirements as he manages meetings along with normal lecturer duties. Another reason that SR was a good information source is that he produced a simple webpage to help himself with his administrative duties with regards to organising the SE subgroup. The interview with JC helped to formalise the process that is normally followed within the SoC when dealing with the distribution of documentation following a meeting.

2.1.3 Information Gathering Techniques

2.1.3.1 A Review of Proven Techniques

After the above users were identified for the initial information extraction, a method of obtaining possible user requirements from them was needed. Therefore, the most appropriate information gathering technique needed to be identified for this stage of information capture. During the research into the varying techniques within this field it became apparent that the most widely used techniques were encompassed by the SQIRO Acronym [2].

2.1.3.2 SQIRO

Information gathering techniques can be split up into five main categories [2, 3], these are outlined below:

1. **S**ample Documents
2. **Q**uestionnaires
3. **I**nterviewing
4. **R**eading (or Research)
5. **O**bservation

These five techniques form the acronym SQIRO. Going through these different techniques systematically; [4] states that "Questionnaires are a cost-effective way to survey a large number of people for facts and opinions on current procedures and possible improvements" and [5] claims that

within interviews the interviewer can act on non-verbal clues, which is not possible with questionnaires. This helped reinforce the author's belief that questionnaires are better for surveying large groups of people and for use when the analyst already has a sound knowledge of what it is that they are modelling. This along with [3] and [6] reiterating the difficulty in constructing a good questionnaire led me to believe that a questionnaire would not be suitable for the information gathering needs. Observation was one possibility, although as the perceived system was going to be a generic prototype MM system, it would not necessarily model the procedures currently being followed by the possible users identified above, this still seemed unsuitable. Observation is normally successful as an information gathering technique when the system being designed is for the user who is being observed [3]. Research was used as an information gathering technique both in terms of identifying the target users, the committees they were involved with, researching existing systems and the MM field in general.

The primary method for information capture was the interview. Reasons for this are as follows; it is the only process for providing sufficient opportunity for the user to air their opinions, and as [5] suggests, "Opinions may be more important and more revealing than facts". [7] shows that interviews are used as a tool to gradually build a model of the system to be implemented. The possible user base was relatively small and this further reinforced the decision to use the interview approach as the primary method for information capture. Sample documents were also identified as a potentially viable information gathering technique and were obtained from the SE website [8] and viewed within the interviews. These sample documents helped to see the structure used for documentation relating to meetings and also helped clarify some different naming conventions used for supporting documents.

2.1.3.3 Conclusion

As stated above, the Interview was the chosen method of primary information capture.
However, various models exist for structuring and carrying out interviews such as the funnel structure, diamond structure etc. [5]. Some of these models seemed suitable for the task of gathering a set of user requirements and the chosen approaches and structure of questions are identified in the following section.

2.1.3.4 Interview Techniques and Structure

It was important to design and structure the interview in such a way as to reap the most information from each target user. The following are just some of the questions that needed answering: Would a structured or unstructured interview be most appropriate? Would open ended or closed questions be the most suitable? etc. The funnel structure [5] seemed the right choice for the first interview as it "provides an easy, non-threatening way to begin an interview". The funnel structure involves starting with a general question and over the course of the interview altering the questions to be more specific, this would hopefully lead to a substantial amount of relevant information. During the conduction of

the three interviews the relevant guidelines outlined in [3] were followed. Questions asked in the interviews were altered in such a way as to try to gain the most information between interviewees of differing technical abilities (in terms of their understanding of the possible benefits of a computerised solution). For example, more direct, technical orientated questions were asked in John Davy's interview than in James Craig's, but a funnel type approach was still used to make the interviewee feel at ease and hence increase the chances of relevant information capture [5]. The interview with JC helped to clarify and document set procedures that are currently followed. This helped to obtain a greater understanding of the administrative tasks associated with organising meetings.

A Dictaphone was used in all except the first interview and the transcripts were later emailed to the interviewees, this 'second look' may help to prompt certain points that had been overlooked during the interview. It also gave the interviewees the opportunity to discuss any discrepancies or to add any further points they had to the transcript. Transcripts of the interviews can be found in Appendix C.

2.1.4 User Requirements

Initially the project schedule was such that the user requirements were to be gathered before the review of existing MM systems. However the gathering of user requirements took place in parallel with both the review of existing systems and relevant background research (see chapter four). The reason for this is that it was decided that these areas compliment each other quite well and hence would increase the success of them. Further changes to the project schedule are discussed in section 6.3. This allowed the author to put forward some possible User Requirements that were generated through reviewing existing systems, background research and individual experience. The initial three interviews and background research provided ample information for a generic prototype system to be produced so the decision was made to begin the design stage of the project and halt gathering further requirements.

The following subsections of 2.1.4 outline the user requirements, firstly those obtained through the interviews and secondly those obtained through the other means described above. These two sections are further divided into those user requirements regarded as essential and those regarded as desirables. Non-Functional user requirements are also mentioned in the latter section.

2.1.4.1 User Requirements Obtained Through Interviews

These user requirements were extracted from the transcripts and characterised into essentials and desirables using the following as a guide:

- If the user requirement was expressed by more than one person throughout the interviews it was deemed an essential.

- If the language used within the interview was such that it was felt that the system needed to provide it then it was also deemed an essential, i.e. if phrases such as “The system should” or “The system would have to do such and such” were used.
- If the language used within the interview was such that it was felt that this functionality would be a nice addition to the system then it was deemed a desirable, i.e. if phrases such as “It would be nice if the system did this” were used.
- Other user requirements extracted that are not classified by the above were deemed desirables.

The initials are given in bold of the interviewee(s) who mentioned the requirement.

Figure 2.1.4.1 – User Requirements obtained via interviews

Essential:
The system shall store documents relating to meetings centrally. **(JD)**
The system shall allow draft documents to be viewed by other committee members. **(JC)**
The system shall specify a mailing list for all members of a certain committee. **(JD, JC)**
The system shall provide a structured template for the production of minutes/agendas. **(JD)**
The system shall allow past minutes and agendas to be viewed depending on search criteria entered. **(JD)**
The system shall allow documents relating to meetings to be viewed by any authorised committee member using an internet browser. **(JD)** [1]
The system shall list the supporting papers relating to each minute/agenda. **(JD, SR)**
The system shall provide supporting papers relating to each minute/agenda if they are available electronically. **(SR, JD)**
The system shall allow past documentation to be retrieved depending on search criteria. **(JD, SR)**
The system shall allow parent-sub committee structure to be recorded. **(SR)**
Desirables:
The system shall allow documents to be fully searchable by content. **(JD)**
The structured template should contain the meeting number and date. **(JD)**
The system shall provide a printed check list for clerical staff which outlines what documents and how much need printing for the meeting in question. **(JD)**
The system shall allow actions to be recorded and managed. **(JD, SR, JC)** [2]
The system shall allow documents relating to meetings to be viewed in different formats. **(JD)**

[1] An Internet browser will be referred to as such, or merely as a browser from this point forward

[2] Although more than one person mentioned this, from the outset this was deemed to be a further enhancement to the system and is hence included on the desirables.

2.1.4.2 User Requirements Obtained Via Other Means

Figure 2.1.4.2 – User Requirements obtained via other means

Essential:
The system shall record various contact details for each person.
The system shall allow new committees to be added.
The system shall record members 'roles' within committees.
The system shall provide protection against unauthorised access.
The system shall list past documents relating to each committee.
The system shall display information relating to the committee including chair, secretary etc.
The system shall hold information relating to committees, i.e. chairs, parent committee etc.
The system shall allow the location of each meeting to be recorded.
The system shall allow meetings to be pre-entered.
The system shall allow meetings to be post-entered.
The system shall allow draft versions of documents to be stored and altered.
Desirables:
The system shall allow password access for alteration of documents.
The system shall allow different access rights for different users.
The system shall allow the default templates for minutes/agendas to be altered.
The system shall run on any operating system that has a graphical internet browser.

The majority of user requirements stated in the previous two figures form the functional requirements that the final system needed to satisfy. Non-functional requirements also needed to be identified and are shown below:

- The system must respond to any request in less than 1.5 seconds.
- The Interface must be easy to use.
- The system should be easy to maintain.

This stage of the project saw completion of minimum requirement two.

2.2 System Requirements

2.2.1 Introduction

The requirements of the proposed system are that it needs to encompass all the essential user requirements as outlined above. In essence, the specification of the end system is merely the list of

essential user requirements. Details of how the system will satisfy these requirements are left to the later chapters involving the design and implementation.

However, the above user requirements have led to some fundamental design decisions that need to be made very early in the project. These decisions are not directly concerned with the appropriate tools, languages or methods used to produce such a system but are concerned with the appropriate approach to undertake for the system produced to be viable. Thus, concern now centres on the appropriate software model to use. Three possible approaches are discussed in the following sections. A number of critical user requirements were extracted from the above section and are then used as criteria to evaluate the model's relevance to the problem at hand, these are shown below:

1. The system shall store documents relating to meetings centrally.
2. The system shall allow documents relating to meetings to be viewed by any authorised committee member using an Internet browser.
3. The system shall provide centralised backup of all documents.

2.2.2 A Standalone System

In the context of this project a standalone system is a system that primarily runs on one PC and only relies on that PC to run i.e. software that can run on one PC and require no other external sources. From the outset this did not seem like a suitable approach as the underlying principles of standalone systems did not lend themselves too well to the fundamental requirements of distributed knowledge/documents that the MM system needed to provide.

2.2.3 A True Client/Server System

A 'True Client/Server system' is a software system that encompasses a client system and a server system. It involves two separate programs; one acting as a server and one acting as a client communicating to each other through proven networking protocols. This approach did seem like a probable candidate for a MM system and could satisfy the user requirements in most places. However the added complexity of designing two systems that communicated in a client/server fashion would make increase the time needed for design and implementation as in reality two separate systems would need to be produced. In a proper software engineering environment added complexities with this approach would exist, such as releasing new versions of the software as all clients would need updating along with the server.

2.2.4 A Web Based System

A web based system can be thought of as a True Client/Server system but one that uses an already completed piece of 3rd party software as the client (i.e. an Internet Browser) and relies on the software written at the server end (although some client side scripting may be present) to supply the required

functionality of the system. Examples of web based technologies used to produce such systems are CGI, ASP etc. This seemed to support the idea of ease of distribution of the related documents and the model seemed the most suited for problem at hand. This model has a major advantage in that if adhered to closely you only need to release a new version of the software on the server and everyone that connects will then gain the benefits.

2.2.5 Conclusion

The above approaches were evaluated against the original list of criteria presented in section 2.2.1 and the results of whether the approach directly supported the user requirements is shown in the table below.

Table 2.2.5 – Evaluation of software models

Criterion	***Standalone***	***Client/Server***	***Web-Based***
1	No	Yes	Yes
2	No	No[3]	Yes
3	No	Yes	Yes

The table suggests that the most suitable approach is that of a web-based approach closely followed by the Client/Server approach. As the web-based approach satisfied all the criteria and because the Client/Server approach would have involved considerably more development, it was decided that a web-based approach would be more suited to this project.

2.3 Storage Requirements

For the system to meet the pre-mentioned requirements, various types of information needed to be stored. Numerous methods could have been used for this purpose from flat text files to complex DBMS's. Whilst flat text files would at first appear to be the simplest to implement, complex I/O routines would need to be developed and performance of the system would deteriorate as the files either got bigger in size or bigger in number. The problem at hand also lent itself very well (as most data does) to the Relational Model and hence a DBMS that was based on the Relational Model would be the most viable solution. Flat text files do not really support the concept of a relational model very well. The storage requirements of the system will therefore need to be satisfied by a DB system, various possible systems are evaluated in 4.3.3.

[3] The client server approach could be modified so that the server side application actually interacted with a web server to encompass the criteria outlined.

The underlying database that will hold the majority of the information for the system will need to be normalised (tables should be in BCNF), adhere to the relational model and provide functionality that includes, but is not limited to, the following:

- Allow authentication
- Allow data to be inserted, updated and deleted.
- Allow relational queries to be executed
- Allow interaction with chosen Programming/Scripting language and technologies.

Information of various types needed to be contained within the DB, this is discussed within the DB design stage which is outlined in section 3.2 and possible DBMS's are discussed in section 4.3.3.

Chapter 3
Design and Methodology

3.1 Possible Methodologies

3.1.1 Introduction

For successful completion of this project and to ensure that a structured, usable and extendable system was developed, it was vital that a suitable methodology was chosen and employed throughout the design and implementation stages of the project. When reviewing various methods it became apparent that most are aimed at large development groups. Due to this they tend to involve a significant amount of documentation and planning for the method itself. A few methods, namely Rapid Application Development (RAD) and methodologies that live under the umbrella of the phrase '*Agile techniques',* were concerned with the rapid development of a usable system, although they normally require a large base of existing software components, from which to construct this system, which due to the nature of the project were not available.

Some key methods are reviewed below; these include The Waterfall Model, The Spiral Model, The Incremental Model, RAD, Agile Techniques and finally RUP. These differing methods all share the common goal of ensuring successful results for the project and encompass requirements gathering, design, development, testing etc. They merely differ by their approach to these areas of software development. For instance the Waterfall Model offers a linear and sequential approach, whereas RAD offers the opposite of a highly iterative approach. Obviously the choice of method will impact severely on the success of the project. For this reason the pre-mentioned methods have been reviewed in greater depth in the sections that follow, a fully informed decision on the most suitable method can then be made.

It is also worth re-iterating that as the system to be produced is generic, it does not really have a distinct problem owner. This means that method's relying on multiple prototypes and continuous feedback from the end user regarding these prototypes, may be inherently un-suitable.

3.1.2 The Waterfall Model

The waterfall model describes a methodology that is linear and sequential [9]. It is one of the most famous (and dated), but it has some serious flaws with regards to its suitability for this project. The user requirements have to be known specifically at the start of the project and well understood [21]. Whilst a preliminary set of user requirements had been gathered, these were by no means complete and other user requirements may come to light during the review of current systems (section 4.2) and

during the design and implementation of the system. This model also assumes a sequential approach to the design and only accommodates iteration in an indirect way [9]. This further detaches its suitability for this project.

3.1.3 The Spiral Model

The spiral model is proven to be a good choice when the requirements change quickly, but it tends to be rather bulky and is more suited to large scale software developments [10]. It also demands that risks be identified and assessed [9], which although useful within large scale mission critical developments is rather a burden on what is required.

3.1.4 The Incremental Model

This is another of the so called Evolutionary Software Process Models [9] and combines the waterfall model with the iterative philosophy of prototyping. A derivative of this approach that is becoming increasingly popular is called eXtreme Programming (although this is not suited as it demands that development is done in pairs). This seemed a likely candidate as it allows an initial system to be developed quickly and tested. This system is then incremented to include more features.

3.1.5 RAD

Rapid Application Development is a "high speed adaptation of the linear sequential model" [9]. It emphasises an extremely short development cycle by cutting down on the initial functionality of the system. However to adhere to this 'fast' development time existing software components are often needed to meet these strict deadlines. This can still be used without existing components being available and may be a possibility as the time constraints of the project are fairly strict. The system can then be further enhanced to include the necessary functionalities.

3.1.6 Agile Techniques

So called 'agile' techniques have become increasingly popular in recent years. These are mainly concerned with four key arguments promoted by the agile Alliance [11].

- Individuals and interactions *over processes and tools.*
- Working software *over comprehensive documentation.*
- Customer collaboration *over contract negotiation.*
- Responding to change *over following a plan.*

Various methodologies exist that follow 'agile' techniques, these include eXtreme Programming, Dynamic Systems Development Method (DSDM) and SCRUM. Various parts of other methods (i.e.

RUP) can be used in an agile way. SCRUM appeared rather appealing when researching these techniques. Schwaber's paper [12] helps provide detailed insight into this empirical method. SCRUM recognises that analysis, design and development stages are unpredictable throughout the software development process and allows for this. The SCRUM Process only has the planning and closure processes of the lifecycle defined. Like most modern Object-Orientated (OO) methodologies SCRUM also assumes a collection of usable modules exist, but it can be used without.

3.1.7 RUP

The Rational Unified Process is one of the most popular methods currently available. It is an iterative approach that ties in well with modelling technologies such as UML. It is dynamic, in that it resides on the internet which means that it should remain up-to-date and it can easily configured to meet different development team's requirements [13]. The major problems with RUP, in terms of this project, are that it is a moderately heavy method and is not strictly suitable for the individual developer who has a relatively short development cycle to adhere to.

3.1.8 Conclusion

The methodology that was needed for this project had to be (amongst other things) flexible, easily adaptable, allow fast implementation of an initial system and iterative (to fit in with the OO paradigm).

It was decided that SCRUM was the most suitable candidate for the development methodology for my project, although only certain aspects of it were used. The main benefit of this method, in terms of this project, is that it allows the initial phase (planning and system architecture) and the last phase (closure) to be defined processes whereas development of modules are done in so called 'sprints'. These 'sprints' allow development, adjustment, wrap and review to be done empirically. A 'sprint' can be thought of as a set of development activities conducted over a pre-defined period. SCRUM allowed the design of the system architecture to be orchestrated initially, but the development and testing of the separate areas of the system can be altered individually throughout the implementation stages. This also allowed smaller milestones to be set for each individual sprint. The author would have to form both the management team and the development team because the project is being undertaken by an individual. Overall this choice of methodology allows quick adaptation and changes to the system when problems arise. A modelling language was also required to allow various design decisions to be visualised and modelled accurately. A decision was made to apply use-case diagrams to model component behaviour and interaction.

Throughout the progression of the project it is important to ensure that the structure of the SCRUM methodology was strictly followed and that it was not cast aside or forgotten as schedule deadlines loom, as avoidance of the proposed methodology when the 'going gets tough' is one of the primary reasons that a lot of software projects fail [11].

3.2 Database Design

The Database that was to form the backend of the system needed to perform a variety of tasks as described in the user requirements. These tasks have been split up into sections relating to storage, input (including updating existing data) and output and are summarised in the following table:

Table 3.2 – DB tasks

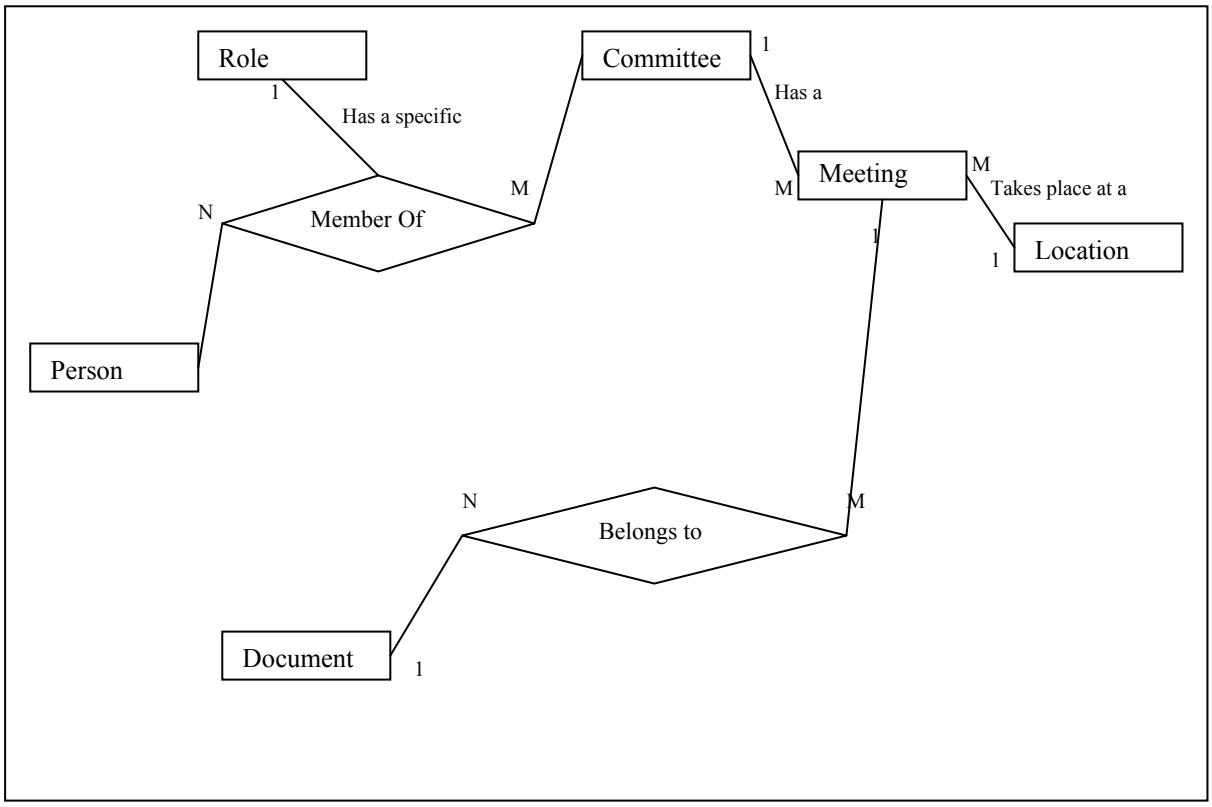
Storage	• Details for each member of a committee • Details for each committee • Details for each meeting • Details for each document belonging to a meeting • Details for each action.
Input	• Allow new members to be added to a committee • Allow new committees to be added • Allow new meetings to be added • Allow new documents to be added • Allow an agenda or minutes to be added to a meeting • Allow criteria to be entered to query the DB and retrieve past documents. • Allow actions to be added.
Output	• Summary, Version and URL's for documents • Meeting details including date, time and location • Documents relating to each meeting. • Committee information including key members of the committee, i.e. chair or secretary and homepage URL. • Email to committee members reminding them of the meeting arrangements [4]

3.2.1 ER-Diagram

The following outlines the original ER-Diagram that the underlying DB was based on. This allowed an initial version of the DB to be quickly derived and implemented. Additional modelling processes were adhered to so that new entities could be added and the functionality of the system increased. A diagram shown in Appendix E helps to depict the final version of the DB.

[4] The email functionality was not envisioned to be part of the DB design but the DB will be used to obtain the email address for the committee members.

Figure 3.2.1 - Initial ER-Diagram for the underlying DB



The above diagram shows two differing relationships that are M:N (many to many), these are *Member Of* and *Belongs to*. The latter refers to the fact that for a particular meeting various documents may exist. This means that within the entities, support must be given to allow a person to be a member of more than one committee and a committee to have more than one member. It also means that the DB Schema needs to model a meeting such that it can have more than one document associated with each meeting. Other essential design assumptions are stated in the next section.

3.2.2 Design Assumptions

For an accurate model of the Universe of Discourse (UoD) to be constructed it follows that various design assumptions needed to be made regarding how these entities interact in the real world. The fundamental design assumptions that correspond to the structure of the DB Schema are stated below in the following figure:

Figure 3.2.2 Fundamental design assumptions for the underlying DB.

- A Person can be a member of more than one committee
- A Committee can have more than one member
- A Person can have differing roles within the different committees.
- A Person's role within the committee is specific to the committee in question.
- A Meeting takes place at one location
- A Meeting must have exactly one or zero minutes and exactly one or zero agenda
- A Meeting can have one, many or zero supporting documents.
- An Action belongs jointly to exactly one person and exactly one committee. [5]
- A Document can be one of three types; a minute, an agenda or a supporting paper.

3.2.3 Relations

Using the above ER diagram it can be noted that two 'many to many' relationships exist and these are shown as diamonds in the ER-diagram, these are both weak entity types [14]. To support this within the DB a table will need to be created.

3.2.4 Entities

The ER-Diagram was analysed and a physical DB design was drafted from the original conceptual schema design. This was later implemented and tested to see if it possessed the desired behavioural aspects. If needed changes were made to the DB (and the system), which allowed the required behaviour to be modelled. The choice of methodology proved very useful here as it allowed for quick adaptation and a re-think in the design of this 'sprint'. The following table outlines the entities that were implemented in the final version of the DB, together with their attributes.

[5] 'Actions' were envisioned to be a further enhancement from the projects conception, but have been included within the original design for completeness.

Table 3.2.4 – The final DB entities and attributes.

Entity	***Attributes***
Action	KeyTable, KeyPerson, KeyCommittee, DateIssued, DateDue, DateCompleted, Summary, Priority, Details, CompletionNotes
Committee	KeyTable, KeyParentCommittee, Name, ShortName, HomepageURL
Document	KeyTable, Type, AbsURL, DtModified, Version, Description
Location	KeyTable, RoomName, Capacity, Description
Meeting	KeyTable, KeyCommittee, KeyLocation, KeyAgenda, KeyMinutes, MeetingNum, Date, EndTime
Member_Of	KeyPerson, KeyCommittee, KeyRole, DtJoined, DtLeftCommittee
Person	KeyTable, Title, FirstName, Surname, EmailAddress, TelNum, HomepageURL
Role	KeyTable, KeyCommittee, Role
Supporting_Docs	KeyMeeting, KeyDocument, DocumentCode

Breaks from the Norm

During the naming of the attributes needed for the Entities extracted from the ER-diagram above, a slightly non standard naming technique was used. This technique was used by the author during his year in industry. It was found that by using this technique, complex multi-table joins could be visualised more efficiently and hence was decided that this approach would help to avoid errors and speed up the implementation phase. This approach is described below, using the Person and Member_Of tables identified:

Normally the primary key of the table corresponding to the Person entity would be named Person_Id (or equivalent). An attribute in the Member_Of table would be Person_Id -to show that a person is a member of a certain committee (which would be represented by Committee_Id, respectively) and hence help model the identified relation. The author's version would have Person_Id renamed to KeyTable in the Person table, and would have every primary Key in every table called KeyTable. Whilst at first glance this appears to be rather questionable these primary keys are then referred to as Key "Table Name" wherever they are referenced as foreign keys (so within the Member_Of table the previous foreign keys, person_Id and Committee_Id would be renamed KeyPerson and KeyCommittee respectively).

This helps to visualise certain multi-table joins as you know that when you a SQL statement to find the names of anyone who has ever been a member of the committee with primary key of 6, you would do:

```
SELECT p.Title, p.FirstName, p.Surname from Person p, Member_Of m
WHERE p.KeyTable = m.KeyPerson
AND m.KeyCommittee = 6
```

When this statement is being devised, you come to doing an inner join on the two tables. You automatically know that for it to be a valid join, one table needs to be joined on its PK and one on its FK. So for it to be valid, one half of the equal sign must be a KeyTable (PK) and one must be Key "*table name*" (FK), where *table name* is the corresponding table relating to the alias preceding KeyTable (hence p in this case is an alias for the table Person and hence one side of the join must be p.KeyTable and the other must be some alias and then join on the KeyPerson attribute of that alias).

3.2.5 Normalisation

To ensure that the underlying DB did not suffer from insert, delete or update anomalies, it was of vital importance to ensure that the DB schema was correctly normalised. To ensure this, functional dependencies, non-trivial dependencies and super keys [14] were extracted and it was shown that the tables were all in Boyce Codd Normal Form (BCNF) or 3rd normal form at the very least using the techniques outlined in [15]. If the tables did not adhere to this then steps would have been taken to decompose the existing relations until they adhered to these rules.

3.2.6 Integrity

Entity and referential integrity were enforced at the DB level [15] using primary key constraints and foreign key constraints respectively. As the DB was designed so that tuples could not be deleted it follows that these constraints should never be violated. The DBMS is responsible for the values of the primary keys (using auto increment or identity seed options) and thus no issues should arise whereby it fails to add a new tuple to the DB due to conflicting key values.

Other constraints needed to be prepared for the system to stop 'mistaken' data from being entered. This included validating forms by ensuring that dates, times etc that were being entered were valid. It was decided to implement constraints that checked input values at the application layer as this helped reduce the load on the DB and provided increased flexibility.

3.2.7 Index Optimisation – A Compromise

When deciding on the best indexing strategy for a DB, it is vital to study the applications use of the DB [15]. This initially posed a problem as the system was experimental and as such, it is not being used properly. This indexing strategy was not initially implemented as that would have involved predicting how the end system was to be used. This was later implemented after the system had been produced and tested.

It was vital to strike a balance between the increased delay on the insertion of new data or the updating of existing data and that of speeding up the selection of the data from the DB (i.e. the DB output). The choice of indexes was based on the author's perceived usage of the system and the differing SQL queries that were being executed were observed. This helped to identify what statements were run most frequently and as such allowed identification of whether more indexes

needed to be added to help speed these particular statements up. The following outlines what indexes were added to the system, although these may need to be revised at a later date to account for increased volume of data.

All tables that contained a primary Key (KeyTable) were given a clustered index on that attribute this is enforced by the DBMS (and is its own way of enforcing the entity integrity rule [15]).

The majority of SQL statements within the application involve an equijoin [14], thus, the default clustered indexes offered significant improvement in the retrieval times. No benefit was to be had by adding or modifying indexes, though this may obviously change as more data is stored and more people use the system. The choice of integer as the data-type for the primary keys also helps to speed up the indexes as each entry in the index will occupy less disk space than say, a string equivalent.

3.3 Program Design

All stages of the design should be abstracted from the technologies to be used, however certain design decisions may not be possible to make until a decision is made with regards to the development platform. A variety of literature, including [16] state that it is good programming practice to separate content from presentation for web-based systems.

3.3.1 Front End

3.3.1.1 Introduction

The Front end is essentially concerned with the presentation aspects of the system. Essentially the user interface would be an advanced web page as it will be displayed through these users internet browser. Web-based solutions often have issues with user interfaces as the interface is the browsers interpretation of the HTML contents received from the server. This can mean a different looking (and acting) interface on Netscape to IE. Different resolutions on the desktop can also cause discrepancies with the user interface to appear in a way not perceived by the designer. For this reason popular desktop resolutions of 800x600 and 1024x768 were tested to ensure that all data would be visible and usable at this depth during the implementation stages.

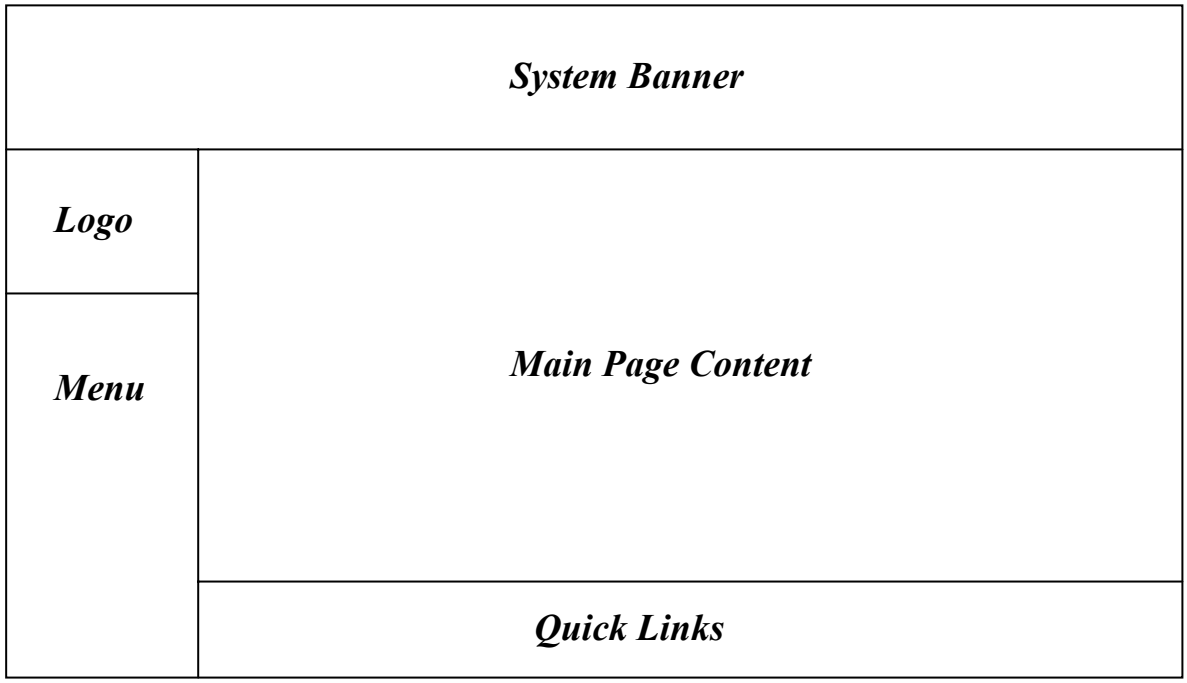
3.3.1.2 HCI Issues

Even though the system to be produced was an experimental system and not meant for commercial use, various considerations still needed to be made regarding HCI and best to avoid potential users having problems with the interface. The interface to be produced followed varying guidelines on how best to structure webpage's including quick links and having a menu to ease navigation on the left. The design principles stated in [17] were also taken into consideration.

3.3.1.3 The GUI

The displayed GUI, is to some degree, enforced by the browser version that the user is running. For this reason, it was decided to not use any browser specific features and to try to stick with simple HTML and HTML forms for input to the system. After researching different examples of web interfaces, a relatively simple approach was used with the emphasis being on consistency throughout all of the pages needed for the system. CCS was used to enforce consistent use of fonts types, size etc. throughout the system. To ensure consistency between the pages they all followed a set template, which is shown below:

Figure 3.3.1- The Template design for each page to adhere to.



3.3.2 Back End

Essentially it was the back end of the system that was responsible for providing the content. This meant that the backend components dealt with inserts and updates to the DB. It is also liable for ensuring that DB connections are closed and that the DB remains in a quiescent state. It was important to design this part of the architecture so as to ensure chunks of code need not be replicated throughout the system.

3.3.2.1 Database Connectivity

The System will need to access data from within the underlying DB. It will need to read this data and modify it. The system will need to ensure that transactions are committed to the DB and that no DB connection remains open for longer than is necessary. The DB connectivity of the system will be

implemented in a separate module so as to ensure that it can be tested properly and changes can be made quickly. As the DB connectivity is dependant on the choice of programming tools it will not be further discussed here.

3.3.2.2 ODBC and DB Drivers

To connect to a DB it is normally required that a driver is available on the residing PC. This allows a program to use method of function calls to access the features of the DBMS. Various types exist, such as Oracle's Oracle Call Interface [16] or Microsoft's Open Database Connectivity (ODBC). Unless the choice of development environment is directly supported by the DBMS, the program is usually required to communicate with these pre-mentioned drivers (or equivalent) in order to access the DB.

3.3.2.3 Uploading Documents

Even though the front end will provide the interface for uploading files, the backend will need to receive the input stream and create the file on the server.

As the system is a web-based one and because documents are to be stored centrally, a means of transferring these documents to the server is needed. If the server was to be stored on the network then a simple method would be to copy the file to a shared directory and then manually enter the absolute URL for the file. However this approach would undoubtedly lead to numerous errors and would impose that the user be of sufficient computer literacy to formulate the correct URL to enter into the system. For these reasons a different approach was needed, the most likely candidate was the use of a HTTP post directive as this protocol will already be in use (due to it being a web-based system). The implementation of this is discussed in section 5.3.2.

3.3.3 Design for Re-Use

As the design was essentially OO, it followed that during the implementation stage a lot of code could be re-used throughout the system using instances of classes or inheritance. This helped to keep the total lines of code down and increase both performance and maintainability of the code. For more details on how the system was wrote to maximise re-use please see section 5.2, as this discusses the matter in more depth and in terms of the specific technologies used.

3.3.4 Possible Security Models

Numerous security models exist that could be applied to this project. Before a decision was made, focus had to be drawn on a variety of issues relating to the system including the on the sensitivity of the data held. Three main issues arise when dealing with security models for a web-based system, these are how secure will the data be in transit?, who will have access to the data? (username/password) and physical security relating to the geography of the server.

As the perceived data to be held was already public available on the SoC intranet it was deemed that sufficient argument was not supplied to enforce that the all data be encrypted from client to server. Implementation of the encryption and decryption routines would have also meant increased development times, and time was not in abundance for the project. Likewise the physical security relating to the geography of the server was not that important as the data did not boast great value to anyone.

However issues did still need to be addressed relating to how people will gain access to the system. Only authorised committee members should be allowed to view the documentation for the meeting and so a security model needed to be implemented. A separate password and username could have been added to each person stored on the DB, this individual user could then use this to log-on and view the documentation. However as this was just a prototype system, and because people do not like to remember another password this was deemed unsuitable.

In the end a decision was made to implement a very simple security model that consisted of two different accounts. One account was 'public' and one was for the 'administrator'. This allowed people to view the details with regards to the meeting location, time etc. and also any documents for the meeting who were logged on in public mode. Whereas the people who are logged on as admin can add new documents, new meetings etc. and use all the other functionality available within the system. It was hoped to improve on this security model (time permitting) and further discussion relating to the choice (and the possible enhancements) are outlined in section 5.3.3**.**

As HTTP is a 'stateless' protocol i.e. it just receives requests and gives responses and has no idea about the previous request from the same client [16] a way was needed for the server to 'remember' the user. Two main possibilities exist, sessions and cookies. The choice for this project was sessions as these offer greater flexibility and are not reliant on the user's browser having them enabled.

Chapter 4

Background Research

4.1 Computerised Meetings Management

Meetings can be more effective by improving a variety of areas around the management and within the actual meeting itself. Meetings can often be deemed a failure and [18] quotes a list of questions to ask (which help show why the meeting failed), the top two reasons for this were "Were the details and requirements of the meeting circulated before hand?" and "Were time location and other participant's names also circulated?" This shows that the distribution of documents and details about the meeting are of utmost importance and fundamental to the meeting being deemed a success. Thus meetings management is a very real issue and improving the process and work flow that surrounds it will inevitably provide committees with more successful and rewarding meetings.

4.1.1 System Necessities

The area of MM is such a wide ranging one encompassing various systems, from document management systems to simple scheduling systems; that it is worth formalising the actual area that concerns this project.

To clarify, the area of meetings management that this project is concerned with is not the scheduling of meetings (as there seems to be a vast array of products that do this, [19] and [20] are just two of the many examples), but the other tasks associated with MM such as the user requirements stated in **2.1.4.** A brief list follows outlining the main characteristics of the system to be produced:

1. Automating the production of certain parts of the Minutes/Agendas.
2. Automating the distribution of various documents that are associated with the meetings.
3. Providing a central storage area for the documents.
4. Allow committee documents to be viewed by any authorised committee member using a browser.
5. Managing Actions (what people have been assigned to do).
6. Providing detailed information on the committee and its members, including homepage URL and email address.
7. Distributing knowledge by emailing committee members with details about the meetings.

4.2 Existing Systems

4.2.1 Introduction

As improved MM can improve the efficiency of the way businesses are run and managed, it follows that commercial interest in such systems is large and hence numerous systems exist. This stage of the project involves a brief review of existing systems in regards to their conformity with the functionality required.

4.2.2 The Current Trend

Research into this field showed that the majority of so called MM systems offered some kind of scheduling service whereby they organise one off events or send you an email reminder on a certain day, but did not concern themselves with the document distribution/management side. Some examples of Virtual project management software were also examined to determine if they had similarities with the one that is required. They did have similarities, but they were just that. The virtual project management software [21], seemed to have some features needed, but lacked the depth required. For instance it allows for pending tasks. Other systems [19, 20 and 22] are primarily concerned with conferences/one off events or just scheduling the time for the meeting and not dealing with the process that happens after a meeting.

However various viable systems did exist but there was no clear de-facto standard. Three of these systems are listed below and are reviewed in the following section.

1. Legistar [23]
2. CoInS [24]
3. modern.gov [25]

4.2.3 Evaluation Criteria

For the systems to be reviewed appropriately a set of criteria needed to be produced to judge their relevance to the stated user requirements. The characteristics stated within section 4.1.1 will be used to evaluate the existing systems as these stemmed from the original user requirements gathered from people within the SoC. These will be referenced by the same number with which they have been listed in section 4.1.1, and an extra row will allow system OS requirements and other proprietary software needed, to be specified for the client. The results of this evaluation are shown in the table below:

Table 4.2 – Evaluation of existing systems

Criterion	***Legistar***	***CoInS***	***modern.gov***
1	Yes, but it is in a proprietary format	No.	Yes, in MS word format
2	Not directly, but documents can be made available to the public via the internet. Manual process.	Distribution Lists allow for this. Although its not stated as automated.	Can publish to the internet, but no mention of letting members know.
3	Not mentioned.	Yes.	Not mentioned, though presumed
4	No mention of authentication via the internet to allow for this.	No mention of authentication via the internet to allow for this.	No mention of authentication via the internet to allow for this.
5	Allows for limited management.	Not supported.	Allows actions to be tracked
6	Not mentioned	Not mentioned.	Not mentioned
7	Not supported directly, though production of a dynamic mailing list can be done.	Not supported directly, but distribution lists would allow for manual email.	Not mentioned.
Sys Req.	Windows 98/NT4/2000/XP	Not mentioned	Windows NT, 2000 or XP & MS Word 2000/XP

4.2.4 Conclusion

The three systems reviewed were mostly council orientated and as such they do bare some similarities with the system required within the SoC. However none of the reviewed systems offered the necessary functionalities to be classed as a solution to the original problem. They are also tied down to the MS architecture and don't support other OS's or technologies. Various specific functions were implemented in these systems that allowed them to meet some key requirements from council users, but abstracted them away from being regarded as a generic system.

The review of these systems brought rise to additional user requirements that would be of perceived use in the final system. This stage of the project oversaw completion of minimum requirement one.

4.3 Possible Technologies

4.3.1 Introduction

To begin implementation of the system is vital that the correct technologies are chosen. The following sections evaluate server side technologies, various DBMS's and client side technologies that may be of benefit during the implementation stage.

4.3.2 An Abundance of Possible Languages

4.3.2.1 Introduction to Server Side Technologies

As the project is now concerned with the production of a web-based MM system, it follows that a major impact of its success will be in choosing the correct Server Side Technology. The software required needs to be developed primarily for execution on the server and then it needs to supply HTML contents to the client machine to satisfy the requirements.

Server Side technologies are sometimes referred to as scripting languages with perhaps the most popular being ASP. The following sections outline four possible choices for this project, before a decision is made on which technology to use.

4.3.2.2 ASP

Microsoft's Active Server Pages are one of the most popular scripting languages used to create dynamic web pages. However the technology has a number of disadvantages to other available technologies. The main disadvantage of this approach is that it because is uses ActiveX controls it is essentially platform specific and hence requires the use of a Microsoft OS [26]. Flavours of ASP are actually available on Linux but none offer the complete functionality (or reliability) of the MS equivalent. It does not offer the portability of JSP or PHP.

4.3.2.3 PHP

Hypertext Preprocessor (PHP – a "recursive acronym") is one of the fastest technologies available and has considerable market share [26]. It is also open source and can run on most platforms. The functionality offered by PHP is also impressive and it would have been exploited if JSP had not been researched. Their were two main reasons that JSP was more viable than PHP, these are that the author already had a knowledge as previous programming experience involved Java and the second was that JSP allows developers to use the full weight of the Java architecture including JDBC, Security, JavaMail etc. which helps with the development and more sophisticated applications. This will help to allow the underlying architecture of the final system to be easy to expand and it will not be likely that language specific restrictions are imposed that may hinder future enhancements.

4.3.2.4 CGI

Common Gateway Interface is a dated technology that is still used in a variety of organisations. It is language-neutral in that it can be used with other programming languages by embedding the CGI scripts. One of the main problems with vanilla CGI is that each CGI script that is run runs as a separate process which leads to performance issues as the number of users accessing the system increases. Complex tasks as also harder to achieve using CGI and the scripts are hard to debug.

4.3.2.5 JSP/Servlets

JavaServer Pages are a relatively new technology that builds on the success of Java Servlets. They support the design approach stated in section 3.3 of separating presentation and content by using JSP pages and JavaBean components respectively. They allow for multi-threading and carry the full weight of the Java architecture. They are highly portable, execute quickly (as it uses compiled classes) and platform independent. JSP pages consist of HTML statements and JSP elements, which in turn consist of directives, actions or scripting elements [16]. This also offered the best choice in terms of expandability as the Java is a very large language that supports numerous technologies and it is always expanding. The benefits of using JSP for this project were all to clear and this confirmed the choice of server side technology to be JSP.

4.3.2.6 Evaluation and Conclusion

As previously stated JSP would be used for the implementation of the system. This meant that a JSP compatible web server needed to be used. The main candidates are Apache Tomcat [27], IBM's WebSphere [28] and BEA's WebLogic [29]. Tomcat was chosen as it is regarded by most as the de-facto standard and it is also free (subject to the Apache Software licence), it was also available within the SoC which further reinforced the belief that this was the server software to be used for the project.

4.3.3 Choice of DBMS

A variety of DB applications exist. Probably the best known DB software is MS Access, this is a general purpose DB system intended for home use and does not really offer the functionality required for large multi-user DB's. MS Access was not considered for this project because it has relatively poor performance and doesn't offer the performance and management tools needed. Another problem with MS Access is that entities cannot relate to each other, hence the KeyParentCommittee attribute of the Committee table (that is a FK to an existing tuples PK within the same table) i.e. the KeyParentCommittee attribute points to an existing Committee tuple could not be modelled.

This left three possible DBMS's that are available within the SoC: MySQL, PostgreSQL and SQL Server 2000 which are reviewed in the following sections.

4.3.3.1 MySQL

One of the main benefits of MySQL is that it is free under the GNU GPL. Binary distributions are also available for both Windows and UNIX, so essentially it is platform independent. MySQL also offers high performance for simple queries though it lacks many features of other high end DBMS's [30].
MySQL would have been an ideal DBMS for the project if it had not been for one major drawback, which was that it does not support active rules.

4.3.3.2 PostgreSQL

The benefits of PostgreSQL is that it is a feature packed DBMS that supports active rules and offers various GUI's [31]. The disadvantages are that it doesn't offer native compatibility with Windows (though this will change in the near future with release PostgreSQL 7.4) and that the interface isn't as easy to use as other DBMS's such as oracle or SQL Server. It is also considerably slower than sMySQL.

4.3.3.3 SQL Server 2000

The benefits of SQL Server are that it provides good online documentation and offers a simple Windows interface [15]. Active rules are also supported. The disadvantages of the DBMS are that it isn't free, it is tied to the Windows platform and it doesn't offer the outright speed of MySQL.

4.3.3.4 Evaluation and Conclusion

Because the underlying DB may require support for active rules, MySQL had to be ruled out. As development was to be done on a Windows platform and because the author has previous experience developing on SQL Server 2000, PostgreSQL would not be used and SQL Server 2000 was used as the DBMS for this project.

If MySQL had supported active rules then this would have been the DBMS of choice as it would have allowed the system to essentially be platform independent and would also have allowed a distribution version of the system to contain the server software (Tomcat) the DBMS software (MySQL) and the MM software. This would then have allowed installation routines to be employed and hence allow the process of software installation to appear more instinctive.

4.3.4 Client Side Technologies

4.3.4.1 The Need for Client Side Validation

New data will be added to the system via HTML forms. It is very important to ensure the integrity of this data before it is stored within the DB. Checking the integrity of the data can also avoid errors and problems of, for example, trying to insert an incorrect date. Client side validation is an extremely useful tool and if used correctly can ensure that the data that is being sent to the server (for storage in

the system) is valid. This can help reduce the load on the server as validation can now be done by the client (with little CPU utilisation) and hence the server will not need to continually send errors back to the client informing them of what is incorrect on the form.

Problems do still exist with this approach, in that most browsers allow you to turn off scripting, hence data will then still need to be validated at the servers end. For this reason, client side validation cannot fully replace the need to check the data at the server end as well. Three different technologies are reviewed below that could be used to provide some form of client side validation.

4.3.4.2 Applets

Applets can be used as a powerful client side validation tool. Applets can also be used to provide a better GUI to the user and offer advanced functionality over static HTML pages. One of the main drawbacks is that for a webpage to run an applet it is required that a JVM is installed on the clients PC. Whilst this is not essentially a big problem as numerous JVM's exist for different platforms and are free, it still imposes an additional requirement for the client PC to adhere to. For this reason Applets will not be used for the system.

4.3.4.3 VBScript

MS's Visual Basic Scripting language is a highly powerful scripting language. It is apparently easy to learn and can provide increased functionality over its competitors as it allows integration with Active X controls. The main draw back with VBScript is that it is tied to the MS platform and IE, Netscape does not support VBScript [32] and as such this does not seem like a likely choice for this project.

4.3.4.4 JavaScript

It is the most widely used scripting language on websites and versions of JavaScript have been supported in browsers as old as Netscape 2.0 and IE 3.0 [33]. Other popular browsers that support this language include Mozilla. IE does not directly support JavaScript, but converts it dynamically into its own version of Jscript [32], so to all sense and purposes it can be thought of supporting it. The language can be hidden from older browsers (that do not support it) and hence will not appear unwontedly as content within the HTML page. It was decided that where necessary JavaScript would be used to provide client side validation. This would be used in a pre-emptive way so as to ensure that the JavaScript validates the data before it is sent to the server and hence avoids increased network traffic, and indeed serves its purpose. JavaScript can be used to create so called 'intelligent forms' that adapt to a past choice on the form, an example of this would be a date drop down list that automatically allocates the correct amount of days for the month chosen so as to avoid an invalid date being entered. This example was used in the final system.

4.3.5 The Suitability of Active Rules

Active rules are a fairly new technology that claims to be able to model business logic and implement business processes effectively. Initially this seemed like a viable technology to use for modelling the general process that committees follow in regards to producing documentation for meetings.

The main source of knowledge on active rules was taken from [34] as this is one of the definitive texts on this topic. Active rules or triggers [35] as they are known in SQL server can be thought of as defining a database's behaviour. This allows the DB to appear intelligent, that it can repair damaged data or provide relevant and meaningful error messages and feedback. A database with active rules defined on it will be referred to as an active database from now on.

A key benefit of Active databases is that they can provide workflow management [35] and [34] states that active rules can be used to define business rules. More specifically workflow managers can be defined as "They support a description of the process, and they help its execution and assignment by monitoring the assignment of tasks to users and by assisting their coordination through document and message exchange." Thus it is clear that Active Databases present a powerful technology that could be adopted for use with the process associated with managing meetings. Active rules are not inherently complex but they can cause a lot of problems when they are combined together with multiple rules on each table within the schema. [34] states that the Commercial systems are rather inadequate at offering a runtime debugger and monitoring software for active rules. Active rules do have a place in a variety of software solutions, however a primary use for them is that of internal integrity management [34] and as such, they would be ideal for a system that had one underlying database and a variety of different front ends so that they could help enforce integrity at the database level.

This project only concerns itself with the production of one interface to the underlying DB and much of what can be done using active rules can be done more efficiently at the application layer. The choice of JSP meant that the full use of the Java API was available and as such it was very likely that most things that could be done in active rules could be done with greater confidence using Java.

4.3.6 Development Tools and Environments

For the project to adhere to the strict temporal constraints, suitable choices of IDE and Development tools are necessary. All development for the system was done on MS Windows XP.

4.3.6.1 SE Environment

Initially all code was written using Vim [36] and then compiled using the default Java compiler from a dos window. However after numerous typographical errors a more suitable choice of IDE was chosen. Borland's JBuilder4 was used to create all of the JavaBeans and also to store the JSP pages. The only way in which JBuilder4 was used in advance of what a text editor was used for is internal compilation

(i.e. the classes could be compiled without the need for a dos prompt) and for the auto-complete functionality which meant that the author could type the first few letters of a method, variable etc, and the IDE would offer a drop down list of the available methods that matched the letters already typed. This helped to avoid unnecessary errors which helped to speed up the actual implementation of the software. HTML was also manually typed within JBuilder4 (as this was contained within the JSP pages).

4.3.6.2 DB Environment

The interface used to develop the DB is normally not a choice, but directly depends on the DBMS that is used. The DBMS of preference was MS SQL Server 2000 and this meant that two main development environments were available. These were SQL Query Analyser and Enterprise Manager. The Former required that the tables, relationships, keys etc. were written using SQL and then executed within Query Analyser to create the tables etc. whilst the latter supplied an GUI that basically allowed you to create the tables/relationships using menus and forms. The latter approach was decided upon as it was thought that this would help to reduce errors and help to speed up the design and implementation of the DB.

Chapter 5

Implementation

The initial schedule was incorrect in that it did not state a design stage within the time frame. This was an initial mistake by the author and the time for implementation needed to be split between design and implementation. For all other changes to the project schedule please see section 6.3.

The following sections outlines the development environment used, good programming techniques adhered to, some of the key problems faced, how they were overcome and major changes made to the design.

5.1 Development Environment

Initially it was hoped to do the development within University or at home and then transfer the program across to the Tomcat server (CSMS7) within the SoC. However numerous problems occurred whilst trying to get a 'driver' version running at University and as such all development was done on the author's individual PC, these problems are outlined in section 5.4.1.

5.2 Good Programming Practice

In following with the Choice of Programming language the OO paradigm was adhered to whenever possible. Good practices presented in the literature on JSP, were used when coding the system. This included allowing the JSP page to concentrate on presenting the content and perhaps formatting input, where as the JavaBean component for that page was responsible for providing the content and/or saving the input to the DB.

Various logically name packages (directories in Java) were created to separate the Java files into the most viable structure. Packages such as 'database' were created that contained the class needed to access the DB (DBConnection.java) and other DB orientated classes, classes in the 'entity' package are concerned with saving (or retrieving) the entities from the DB.

It is regarded as good practice to implement a Java class to access the DB rather than implementing the required functions within a JSP page [16]. The other JavaBeans then used an instance of this class to establish a DB connection, retrieve the required information or update/insert the data and then close the connection. Another class within the database package, DBQuery, was designed as a helper class that could be used by the other JavaBeans to easily and accurately create complex SQL statements. The architecture was designed in this way so as to emphasise code re-use and also to enable minimal code change to support a different DBMS.

Most JSP pages had corresponding JavaBeans components and they followed a sensible naming convention in that the comOverview.jsp page would rely on a JavaBean called

comOverviewBean.java. The JSP tags <jsp:useBean> and <jsp:setProperty> were used extensively to allow the properties of the JavaBean component to be set correctly before being saved to the DB. The use of the JavaBeans component architecture allowed for numerous code re-use by allowing existing components to be repeatedly used on differing JSP pages. An example of this was the EmailBean class which was re-used for three separate JSP pages.

5.3 Summary of Development

The first stage of the development process was to write a simple driver program to access the underlying DB. After this was complete and it was established that the Tomcat server was correctly configured a template had to be created that all other JSP pages would use, a screen dump of the template is shown below:

Figure 5.3 – Template used for each page of the System

Meetings Management - the central administrative solution - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://81.96.85.137:8080/testLogin.jsp
Go Links
Meetings Management
Home
New Person
New Committee
New Com Member
Log In
Main page content goes here
Home | New Person | New Committee | Add a Committee Member
© Josh O'Neill 2003

5.3.1 Page Implementation

After the template was created a systematic approach was adopted by ensuring that individual functionalities required of the system were addressed singularly. The approach used was that of designing the JSP page required and then the JavaBeans component for that page. Validation was

added to the JSP pages throughout the development either through the JSP page itself or using JavaScript (or both). This was done originally for each entity within the DB design, then for the pages offering email functionality and then a central page had to be written that would make all this information available to the user. This was essentially the home page of the system and may be referred to as such from now on; the term 'main page' may also be used. A screen dump is shown in Appendix D.

5.3.2 Uploading Documents to the Server

The approach that was implemented, used the HTML <input type="File"> form tag which allows users to choose a file from their PC to upload and ensuring that the form uses the POST method (so it sends data to the server) and an enctype of MULTIPART/FORM-DATA (which allows the client to send file data to the server over the stream). This stream still needed to be handled at the server end and a file needed to be created (containing the contents of this stream). The ServletInputStream class from the javax API was used to do this. This was created as a JavaBean to again allow the component to be re-used should file transfer be required anywhere else within the system.

5.3.3 Security Model

The Security model was relatively easy to implement, it involved the creation of a table to store the details for the user, i.e. username and password and a JSP page to allow users to logon. This JSP page would check the input against the stored values for username and password and if it was correct it would allow the user to logon. The type of user that was logged on was then added to the current session and every page in the system had to check the session to see if the user was allowed access the page they were trying to. This helped stop public users from adding new documents to the system or emailing committee members.

5.4 Problems Encountered

5.4.1 Connection Issues

One of the main problems during the early stages of development was due to problems trying to obtain a DB connection to the server within the SoC using a JSP page. These problems initially stemmed from a suitable JDBC driver not being installed on the DB server. A standalone application was written that could access the DB from another computer within the SoC. The DB could also be accessed by physically running this standalone application from the actual server (CSMS7). However DB access could not be established when tried via a JSP page within Tomcat. Trying to get a simple DB connection using JSP within the SoC proved rather difficult and a week was spent on this issue

which unavoidably caused delays to the project schedule. For details on the changes brought about to the project schedule please see section 6.3.

5.4.2 Tomcat Configuration

Due to the problems outlined in the last section a usable JSP compatible server was not available for the project. This meant that Tomcat had to be installed and configured on the author's PC so that implementation wouldn't be further delayed. This was a considerable task as no previous experience of web-servers had been obtained. However, Tomcat was configured correctly and a DB connection was established. [37] provided a vital source of knowledge when issues relating to the configuration of Tomcat cam to light. The lessons learnt here were also used again on the SoC server to see if the connection issues could be fixed, but to no avail.

5.4.3 Issues Relating to Design

Throughout the implementation stages various issues cropped up which meant that the design needed to be changed in some way. Some of these were oversights with the original design; some were just related to supplying the user needs more directly. The choice of methodology helped to allow redesign, testing and evaluation of modules to be carried out with the minimum outlay of time.
The following outlines some of the major changes to the original design.

Initially the system only allowed a set of predefined roles (i.e. chair) to be chosen for a member of a committee. A new entity was created that allowed new roles to be added to the system. These roles would be specific to the committee to which they are added.

Initially the system would display 'Supporting Document' for the hyperlink to the supporting document for a meeting. It was mentioned in the progress meeting that it would be better if it displayed the document code (as most supporting papers follow a set coding scheme). This meant that an extra attribute (DocumentCode) needed adding to the table Supporting_Docs and minor alterations made to the JSP page.

Email functionality was added to the system. Individuals could then be reminded about outstanding actions, informed of the details concerning a meeting or informed that new documents had been added for the meeting.

The Interface was redesigned to allow the main page content to remain central at all times, a more suitable style sheet was also designed to offer greater support in differing browsers.

A conscious decision was made to not allow users to amend some existing entities directly. The reason for this is too avoid classic temporal DB issues such as if a person renames a committee to a different name. For example, if a new committee has started and the committee to be renamed is disband then all the related tuples now belong to this perceived new committee and hence it looks as though this new committee has had numerous meetings and has numerous members even though it doesn't. This approach will help to avoid these issues and help to avoid compromising the data's integrity.

The search functionality of the system is rather limited in that only certain search criteria can be used. A version of the search page was implemented that offered full content addressable searching which allowed users to find files that contained any string within the files contents. It also supplied results to searches on attributes such as file size, document author and when the file was modified. It used MS's Indexing Service which is included in Windows 2000/XP to achieve the content addressable search. This advanced search functionality has not been included in the final system as it had some issues with regards to reliability and with the firewall software on the server. Given more time the functionality could have been implemented correctly and released in the final version of the software.

Chapter 6

Evaluation

For the project to be deemed a success, it is of vital importance that it met its original objectives and satisfied the minimum requirements.

This stage of the project has already overseen completion of minimum requirements one and two which were completed within section 4.2 and 2.1 respectively. Chapter five provided an overview of the implementation of the MM system (which goes some way to satisfying minimum requirement three). This chapter is concerned with validating that a generic prototype MM system has been produced (min requirement three) and that it satisfies the essential user requirements. It is also concerned with establishing that the objectives were met and that the solution is evaluated accurately (which in turn completes the last minimum requirement).

6.1 Testing of the MM System

When new additions to the system were made it was vital to ensure that these were tested adequately. The choice of methodology allowed the system to be broken into smaller modules (sprints), which in turn meant that testing could be carried out in smaller more frequent steps. Issues could then be addressed quickly and steps taken to ensure similar problems did not arise again during the implementation.

As the system was a prototype, various aspects needed to be improved (in terms of increased validation and user feedback), before it could be used with the required level of confidence in the commercial field. The following sections briefly summarise the testing that was carried out on the system. It should be noted that the separate modules were tested individually before assessing their combined interaction.

6.1.1 Input Validation

Input validation was necessary to ensure that no incorrect data was entered into the DB and no errors occurred when new information was added to the system. The techniques used to implement this have been described in section 4.3.4. As the system was a prototype, advanced input validation was not implemented, for instance, checking the validity of inputted email addresses was not available. However the system did check for correct dates, and correct times entered into the system. In checking for correct dates, JavaScript was used to dynamically create the drop down lists. It ensured that the correct range of days was specified depending on the choice of month, or year.

Times were verified using JSP. These validation routines should both be executed at the client side and the server side for the reasons discussed in section 4.3.4.4. Various improvements could be made in regards to validating all input and this is discussed in section 7.1.

6.1.2 Performance

Various measurements were taken to ensure that the level of performance of the system was acceptable. To evaluate the performance of the system effectively it was necessary to connect to the system using an external PC, and then time the response times/time taken to load each page. The time taken to load a page had other uncontrollable external factors such as network/internet traffic at that time, the download speed of the client, and the upload speed of the server. Overall, each page still took less than 1.5 seconds to load when the server was in development mode. Development mode allows the server to detect and compile JSP pages that have been modified after the JSP page that is currently stored in memory. There is no need to stop and restart the Tomcat service [37]. Development inevitably degraded performance and was subsequently turned off (as it should be for deployment situations). This increased performance to such that no page failed to load in less than one second.

6.1.3 Document Upload

It was essential to ensure that the document upload worked flawlessly. Various documents were uploaded including MS word format, HTML and plain text files. The document upload component of the system also updated existing data within the DB by, for example, updating the KeyAgenda attribute of the meeting tuple if a new agenda was being uploaded. For this reason it was vital to ensure that the data's integrity was not compromised. This often involved manually running some SQL to ensure that the system was behaving correctly and that the correct tuple's were being updated with the correct value.

6.1.4 Security

Simple testing was carried out to ensure that the security model behaved as it should. Attempts were made to type in the absolute URL of pages within the system to determine if such knowledge would allow them to be viewed without first logging into the system. All pages will forward the user to the logon page, thereby prompting for authentication details. The system was then tested with the user logged on in public mode. Attempts were made to access the other pages, for example those allowing documents to be uploaded or committee members to be emailed. One page was found to unintentionally allow public access; this was immediately fixed and re-tested. Attempts were also made to login with an incorrect username and password combination. The system always refused entry and provided feedback to the user informing them that the authentication details were invalid and prompted for re-entry.

6.2 Evaluation of the MM System

For the successful completion of minimum requirement four, "Produce a prototype generic MM system" it is important that the required functionality of the system is shown to have been met. The following figure outlines the initial essential user requirements (i.e. the functional requirements), and shows whether the end system met those requirements. As the system had to satisfy both these requirements and the non-functional requirements stated for the system, it follows that this area would form a good starting post for the evaluation of the system to evolve from. To deliver the third minimum requirement it was previously stated that the system would have to offer all the functionalities that were classified as essential. For this reason the following sections have been separated to encompass how the system met the user requirements regarded as essential, and how it met the desirables (i.e. how it surpassed the minimum requirements by producing these to a higher standard or by offering enhanced functionality). A brief overview is presented on the entire system comparing it to existing systems and providing potential user feedback. The potential for expandability of the system is also discussed before the whole evaluation is concluded.

6.2.1 Essential User Requirements

To regard the project as a success and to verify that the minimum requirement three had been delivered, it was fundamental that the system met the essential user requirements laid out in section 2.1.4.

Table 6.2.1.1 – Evaluation of system against essential user requirements from interviews.

User Requirement	***Solution offered by the system***
The system shall store documents relating to meetings centrally. **(JD)**	Yes this is supported.
The system shall allow draft documents to be viewed by other committee members. **(JC)**	Yes this is supported.
The system shall specify a mailing list for all members of a certain committee. **(JD, JC)**	Yes this is supported.
The system shall provide a structured template for the production of minutes/agendas. **(JD)**	Supported, the template produced is dynamic and contains meeting date, number etc
The system shall allow past minutes and agendas to be viewed depending on search criteria entered. **(JD)**	A limited search page is available and the last 10 meetings and links to corresponding documents are shown on main page.
The system shall allow documents relating to meetings to be viewed by any authorised committee member using an internet browser.	Yes this is supported, public users can view these documents using a valid user name and password

(JD)	
The system shall list the supporting papers relating to each minute/agenda. **(JD, SR)**	Yes this is supported on the main page, a supporting document code can also be entered
The system shall provide supporting papers relating to each minute/agenda if they are available electronically. **(SR, JD)**	The documents can be uploaded to the system and a hyperlink is automatically generated.
The system shall allow parent-sub committee structure to be recorded. **(SR)**	This is available when adding a new committee, simply chose the correct parent committee (NB: the parent committee must already exist on the system.

As can be seen from the above, the system satisfied the essential user requirements obtained via interview. However those obtained via other means needed to be satisfied. The results of this are shown below:

Table 6.2.1.2 – Evaluation of system against user requirements obtained via other means.

User Requirement	***Solution offered by the system***
The system shall record various contact details for each person.	Yes, email address, telephone number and homepage URL can be recorded for each person.
The system shall allow new committees to be added.	Yes, this can be done using the menu on the left.
The system shall record members 'roles' within committees.	Yes, when adding a new member simple choose the correct roles. New roles can be added which are committee specific.
The system shall provide protection against unauthorised access.	A correct username and password are required to gain entry,
The system shall list past documents relating to each committee.	This is achieved on the main screen which shows past meetings and corresponding documentation.
The system shall display information relating to the committee including chair, secretary etc.	This is also done on the main page.
The system shall hold information relating to committees, i.e. chairs, parent committee etc.	The underlying DB design allows for this.
The system shall allow the location of each meeting to be recorded.	Yes this is possible, new locations can also be added to the system.
The system shall allow meetings to be pre-entered.	Yes this is encouraged, an automated email can be sent out informing committee members.

The system shall allow meetings to be post-entered.	Support is available for this.
The system shall allow draft versions of documents to be stored and altered.	The system allows draft versions to be stored. These can then be downloaded, altered, and re-submitted to the system hence modelling the process.

The Non-Functional user requirements also needed to be satisfied. Due to their definition it is fundamentally hard to show that Non-Functional requirements have been met and this project was certainly no exception. The first of these concerned the systems response times. This was shown to have been satisfied whilst testing was being carried out on the performance of the system in section 6.1.2. The potential users commented on the interface and stated that whilst it is not the most sophisticated interface they have seen it was certainly easy to use. It therefore satisfied the second Non-Functional requirement requesting that the interface be easy to use. The final Non-Functional requirement was satisfied in that the maintenance of the system was relatively easy to carry out. This simply involved creating the necessary directory for any new committee.

6.2.2 Desirable User Requirements and Improvements

In addition to meeting the essential user requirements outlined above the system also satisfied the following desirables:

- The structured template should contain the meeting number and date. **(JD)**
- The system shall allow actions to be recorded and managed. **(JD, SR, JC)**
- The system shall allow password access for alteration of documents.
- The system shall run on any operating system that has a graphical internet browser.
- The system shall allow different access rights for different users.

The last requirement was not met completely because the system did not sufficiently offer different access rights for different users. The current model had two different sets of user, those regarded as public, and those regarded as administrators. These different roles could subsequently gain access to different areas of the system.

6.2.3 The End System

6.2.3.1 Overview

What follows is a brief list of some of the main functionalities offered by the system:

- Lists the last 10 meetings for the committee including dynamic links to all electronically available documentation for each meeting.
- Allows dynamic creation of cross platform friendly templates for both the meetings agenda and minutes (by creating documents in a widely support format).
- Allows for management of tasks including clearly outlining the outstanding tasks associated with each committee, including emailing reminders to task owners.
- Provides a central area for document storage so problems such as members receiving out-of-date or old versions of minutes/agendas is likely to occur. Centralised backups of the software can also be carried out easily.
- Shows the key members of each committee, such as Chair or Secretary, and provides contact information including homepage URL if available. Can also produce a mailing list for the entire committee.
- Allows any document to be uploaded and related to the meeting as a minute, agenda or Supporting paper.
- Allows automated emails to committee members informing them of a meeting that has been organised including date and time etc.
- Provides automated emails allowing committee members to be emailed informing them that some new, or updated documentation has been added to the system, and provides direct URL's to this documentation.
- Supports draft versions of documents.

6.2.3.2 Comparison to Existing Systems

As the overview above shows, the prototype system produced offered some genuinely useful functions. To produce a fair comparison of this system to existing systems, it is important that the same criteria is re-used. The following table outlines the same criteria presented in section 4.2.3 and the solution offered by the final system.

Table 6.2.3.2 – Evaluation of system using criteria defined in section 4.2.3

Criterion	***Solution offered by the system***
1	Yes, allows template to be created dynamically in HTML.
2	Yes Allows links to be presented on the home page and emails to be sent to committee members with direct URL's to all the latest files.
3	Yes all documents are stored in a logical file structure on the DB.
4	Yes allows authentication to be verified and documents to be accessed if the user has the correct username and password.
5	Supports management of tasks and allows reminder emails to be sent.

6	Provides information on the committee and its members, including those requested.
7	Allows emails to be sent to committee members informing them of a meeting that has been organised.
Sys Req.	Any OS: with an Internet Browser

As can be seen from the above table the system meets all of the set criteria. This should not really be surprising as the system was designed to meet the user requirements gathered; and this set of criteria stemmed from the original requirements. What it does show is that in terms of the SoC the prototype system implemented for this project would be of greater benefit than the existing systems reviewed.

6.2.3.3 Security Model

From the outset a simple approach was taken for the security model to be used. However various models exist that would be of more use as the number of system users increased. Perhaps one of the best models to use within the SoC would be to store the server on an intranet and allow certain members administrative access to the system using their SoC credentials. The SoC credentials could be used against their specific role, and which committees they are members of, to allow them to have different access rights to the various committees. This would also avoid the need to remember another password (and the issues that follow associated with password maintenance). Careful consideration would need to be employed here as committees can also distribute sensitive information. Steps would have to be taken to ensure that this did not fall into the wrong hands. Various data protection acts would also need to be researched for this to be used in a deployment environment.

6.2.3.4 Potential User Feedback

To critically appraise the system it was important to get some feedback from potential beneficiaries within the SoC. Both JD and SAR were approached to put forward their views on the final system. The feedback was generally positive and helped to pinpoint the main weaknesses of the system. It also allowed those critical in producing the set of user requirements to review the system and contemplate its benefits and further enhancements. It was thought by JD that "It would be of benefit, but it would require use by all committee members". SAR commented that "It would definitely be an improvement to the current paper based system used".

6.2.3.5 Expandability

The project set out as essentially an experimental one into the area of MM. It involved the production of a generic system and not one slanted towards an individual, or company's requirements. As the project progressed it became increasingly obvious, that the possibilities of the system are seemingly endless. The choice of development environment being JSP and Java added to this, as Java is a large language and offers most things that are computationally possible. Employing a flexible strategy

allowed many of the further enhancements to come to light during the design and implementation stages. Some of the most useful possibilities in terms of future enhancements are mentioned in section 7.1.

6.2.4 Conclusion

"The problem to be solved is that of improving the organisation and distribution of documents to make meetings more efficient". (section 1.1)

The original problem was stated in such a way as to allow this question to be answered in a straight forward manner. The final system would improve on the current paper based and could be used with confidence within the SoC. Feedback from potential users reinforces this belief. It is worth noting at this stage that for the system to be used as a commercial tool within the SoC then a different license would be needed for SQL Server 2000. However, as the system does not currently use active rules it would be relatively easy to incorporate support for a different DBMS which is discussed as a future enhancement in section 7.1.

6.3 Evaluation of the Project Schedule/Plan

Various issues arose throughout the project lifecycle that caused some sort of change to the original schedule. These ranged from the unavoidable (such as the progress meeting that relies on the scheduling of three individuals timetables) to the unforeseen (a few key tasks should have been planned to be executed in parallel). Overall the project plan was deemed a success as it allowed for completion of the project and provided sufficient time for writing it up.

A few flaws with the project plan were identified during the completion of the projects lifecycle. These were that it presumed certain dates would be met, even though reliance was directed towards third parties. An example of this was the progress meeting where the original project plan merely stated one day in the schedule that would be used for the progress meeting. In reality this was very unlikely to happen, as three schedules needed to be checked and a mutually available time had to be booked. Another flaw with the schedule was that it did not identify tasks that would gain benefit by being completed together. This was identified and rectified during the project by allowing user requirements and a review of existing systems to be done in parallel. Allowances were not given to the schedule for such things as Environment configuration or issues involving the configuration of Tomcat. A week was spent at University trying to correctly configure Tomcat between March 10th and 17th, to no avail.

However, despite these shortfalls in the original schedule, the author remained in control of the project by documenting and modifying the schedule when these problems occurred. Due to a lot of deliverables specified to be completed within relatively long time frames it allowed the schedule to be 'caught' up by working longer hours than planned.

Various problems occurred during the project when trying to organise meetings with potential beneficiaries within the SoC, these ranged from Jury duty, away with long term sickness or just had a change of responsibilities within their job. This meant that the final set of user requirements wasn't actually formulated until 28th February, and as such this caused a delay in reviewing existing systems. However the initial design stage could still be started as sufficient knowledge was acquired during the first interview and subsequent background research.

Despite all these issues, the project schedule and management side of the project was still deemed a success because it needed to strike a balance between remaining fairly simple, whilst still allowing sufficient problems to be modelled to ensure the project accomplished what was required. The choice of methodology also played a key role in the projects success as it allowed for changes to easily be made in terms of time frames and deadlines.

The alteration of deadlines, changes to the original project schedule and corresponding Gantt charts are shown in Appendix B.

Chapter 7

Conclusion and Future Enhancements

The project could be regarded as experimental; in that it was not attempting to solve a distinct problem owned by a distinct problem owner. Instead it was involved with the creation of a generic prototype MM system and the benefits it could offer to the SoC. This approach does not bring to light all future enhancements after the analysis but during the design, implementation and evaluation of the system.

The nature of the system also meant that the scope for future enhancements was of great magnitude. The existing system should not be seen as a release ready project, but more as an experimental MM system that can act as a base for others to build on. This chapter outlines possible future enhancements that would benefit the system. It also provides hints on how they could be implemented, before finally concluding the project and the report.

7.1 Future Developments

As mentioned at the end of section 5.4.3, a fully content addressable searching system would be a nice addition to the system and would offer real benefit to potential users. Various methods could be used to implement this. The file could be stored within the DB (either by inserting it directly –if its plain text or HTML, or by converting it first). A more superior approach would be the one attempted by the author, of using IS to index the file and help to provide this for you. Files can then retain their original form whilst still supporting content searching, hence word documents, PDF files etc. can be searched.

A current weakness of the system is that the exception and error handling routines are rather basic. Java offers extensive error handling and use should be made of this. If it was not for the temporal constraints of the project then improved error handling would have already been implemented. JSP also offers good error handling. They allow pages to be set up to act as error pages for another JSP page to use. This basically stops the user from seeing errors when using they occur, whilst at the same time recording the error and allowing the developer to see details of the cause.

As each separate item on the minutes normally has a distinct number, it follows that these could be modelled conceptually in such a way as to allow them to be stored separately. These could then be dynamically referenced within future documents. The system could then allow particular items to be traced back from their inception, to their implementation or completeness. This would allow the decision process to be tracked back and analysed. Probably the most appropriate way to do this would be to implement specific minute items using an XML template with correctly defined tags.

Another item of interest from [38] mentions the use of XML style sheets (XSL) which allows XML enabled documents to be published as HTML, PDF or in a variety of other formats. A possible enhancement to the system would be to store the minutes/agenda in XML (as mentioned in previous enhancement) and then allow a user of the system to specify their favourite document viewing format. Whilst at first this may appear to be a luxury it would avoid problems of cross platform support such as Unix users trying to view MS word documents. This could be implemented by storing a cookie on the client's PC that tells the system which document format they prefer. It would be relatively straightforward to implement XSL within the system as Java offers substantial XML support.

Some maintenance is required server side for every time a new committee is added to the system. A directory has to be created for each new committee that is added to the system (this directory is named to whatever value the committee's PK is). This could easily be changed so that every time a new committee is added to the system a directory is created using the File class from the Java IO package. Good examples of how straightforward this would be to implement can be found in [39].

The security model implemented is not suited for use with a commercial system. A future enhancement would be to offer a different, more advanced security model. Section 6.2.3.3 gives some pointers on possible security model improvements.

Numerous improvements could be made to the user interface. Potential enhancements to this area of the system include increased validation and user feedback outlining exactly what the error is on the form. Increased validation would include ensuring that dates, times and email addresses are validated firstly on the client and then on the server – in case client side scripting is not available. These could be implemented using JavaScript for the client side and JSP for the server side. For example, times could be validated in JavaScript by specifying a regular expression that models the accepted time format. It would then check if a string containing the contents of the time field of the form adhered to the regular expression, by using the match() method of the String class [33].

As mentioned in section 6.2.4, the current University licence for SQL Server would not allow the system to be used in a commercial way. An obvious enhancement would be to allow the system to support another DBMS. It could even be made to run in various modes so as to allow the person who installs the system to use the DBMS they are most confident with or the one they already have installed. MySQL could be supported as the final system does not use active rules. This would be a straightforward further enhancement and would simply involve the alteration of around 5-10 lines of code within the /fyproj/database/DBConnection.java file (which is in charge of managing the DB connections). A few changes might be needed throughout the system; as a few SQL Server specific keywords were essential to offer the required functionality. Other DBMS's have similar functionality

but just use different keywords. The only SQL Server specific keywords used were TOP and CONVERT.

7.2 Project Conclusion

The discussion presented in chapter six showed the project to be a success. All aims of the project were reached throughout the duration of the project and the minimum requirements were met to a high standard. Due to the nature of the project, the scope for future work was vast. Possible enhancements have been outlined in the previous section, along with suggested approaches on how best to implement them.

Bibliography

[1] (1997) **Chambers Concise Dictionary,** Chambers Harrap Publishers Ltd.

[2] Johnson, Owen, *Internet Systems Engineering*, URL:http://www.comp.leeds.ac.uk/in21/slides.html [10th December 2002]

[3] Bennet, S, McRobb, S, Farmer R (1999), *Requirements Capture*, in: **Object-Orientated Systems Analysis and Design using UML**, pp. 96-119, McGraw-Hill.

[4] Kendall, Penny A, (1992), *Fact-Gathering Techniques*, in: **Introductions to Systems Analysis and Design: A Structured Approach**, Second Edition, pp. 65-89, Wm. C. Brown.Ds.

[5] Kendall, KE, Kendall, JE (1999), *Interviewing*, in: **Systems Analysis and Design**, Fourth Edition, pp. 111-169, Prentice Hall.

[6] Daniels, A, Yeates, D (1984), **Basic Systems Analysis**, Second Edition, pp. 13-19, Pitman Publishing Ltd.

[7] Hawryszkiewycz, I.T, (1988), **Introduction to Systems Analysis and Design**, pp.44-47, Prentice Hall.

[8] (2002), SE Review Homepage, URL:http://www.db.comp.leeds.ac.uk/sar/SE_Review/index.asp [12th March 2003]

[9] Pressman, Roger S., (1997), **Software Engineering A Practitioner's Approach**, Fourth Edition, pp.22-56, McGraw-Hill.

[10] Sommerville, Ian (2001), **Software Engineering**, Sixth Edition, Addison-Wesley.

[11] (2003), *Agile Alliance*, URL:http://www.agilealliance.com/ [26th Jan 2003]

[12] (2003), *SCRUM Development Process*, URL:http://jeffsutherland.com/oopsla/schwapub.pdf [24th Jan 2003]

[13] (2001), *IN22 Homepage,*URL:http://www.comp.leeds.ac.uk/in22/ [02nd March 2003]

[14] Elmasri R, Navathe S, (2000), **Fundamentals of Database Systems**, International Edition, Addison-Wesley.

[15] Roberts, Stuart, (2002), *DB21: Database Principles and Practice Module Notes 2002-3*, The University Of Leeds.

[16] Hanna, Phil, (2003), **JSP 2.0 The Complete Reference**, Mc-Graw-Hill/Osborne.

[17] Dix, Alan, Finlay, Janet, Abowd, Gregory, Beale, Russel, (1998), **Human-Computer Interaction**, Second Edition, p.594-610, Prentice Hall

[18] Hodgson, Philip, Hodgson Jane, (1992), **Effective Meetings**, pp20-31, Century Business.

[19] (2002), *Meeting Wizard*, URL:http://www.meetingwizard.com/mwiz/home/default.cfm [10th Dec 2002]

[20] (2002), *Office Tracker*, URL:http://www.milum.com/ [10th Dec 2002]

[21] (2002), *Virtual Project Manager*, URL:http://www.virtual-pm.com/ [10th Dec 2002]

[22] (2002), *Event Management Systems*, URL:http://www.dea.com/ [10th Dec 2002]

[23] (1998), *Daystar Computer Systems*, URL: http://www.daystarnet.com [5th Feb 2003]

[24] (2002), *Anite public Sector- Council Information System (CoInS)*, URL:http://www.aniteps.com/products/coins.html [7th Feb 2003]

[25] (2002), *modern.*gov, URL:http://www.moderngov.co.uk/ [11th Feb 2003]

[26] (2003), *The Infrastructure of the WWW – Dr K Chen*, URL:http://homepages.wmich.edu/~kchen/courses/bis490/The%20Infrastructure%20of%20the%20WWW.ppt [19th Feb 2003]

[27] (2003), *Apache Tomcat*, URL:http://jakarta.apache.org/tomcat/ [12th March 2003]

[28] (2003), *IBM WebSphere Application Server*, URL:http://www-4.ibm.com/software/webservers/appserv/ [1st March 2003]

[29] (2003), *BEA WebLogic Server*, URL:http://www.beasys.com/framework.jsp?CNT=index.htm&FP=/content/products/server [3rd March 2003]

[30] (2001), *Notes on software for dynamic* websites, URL:http://libweb.uoregon.edu/diglib/JQJ-Dynamic-Web-Sites.html [18th Feb 2003]

[31] (2003), *PostgreSQL Homepage*, URL:http://www.postgresql.org/ [20th Feb 2003]

[32] (2003), *VBScript*, URL:http://searchwin2000.techtarget.com/sDefinition/0,,sid1_gci213279,00.html [23rd March 2003]

[33] Flanagan, David, (2003), **JavaScript Pocket Reference**, 2nd Edition, O'Reilly & Associates Inc.

[34] Zaniolo C, Ceri S, Faloutsos C, Snodgrass RT, Subrahmanian VS and Zicari R, (1997), *Part 1: Active Databases*, in: **Advanced Database Systems**, pp.5-95, Morgan Kaufmann

[35] Roberts, Stuart, (2002), *DB31: Advanced Databases Module Notes 2002-3 PART 1: Active Databases*, The University Of Leeds.

[36] (2003), Vim the editor, URL:http://www.vim.org/ [12th Jan 2003]

[37] (2003), *Configuring & Using Apache Tomcat 4*, URL:http://www.moreservlets.com/Using-Tomcat-4.html [23rd April 2003]

[38] Schmelzer, R, Vandersypen, T, Bloomberg, J, Siddalingaiah, M, Hunting, S, Qualls, MD, Houlding, D, Darby, C, Kennedy, D, (2002), **XML and Web Services Unleashed**, pp.370-380, Sams Publishing.

[39] (1997), *Contents of Doc 18 Java IO: File, Stream Tokenizer*, URL:http://www.eli.sdsu.edu/courses/fall97/cs535/notes/javaIO2/javaIO2.html#RTFToC3 [23rd April 2003]

Appendix A

Reflection

During my completion of the Computer Science degree within the University of Leeds, I have been challenged by extensive coursework and complex modules. However, no past experience could prepare me for the amount of work needed to produce a project of sufficient quality. Having said that, I thoroughly enjoyed certain aspects of the project and I am confident that the system provided met and surpassed the original objectives and requirements. I was pleased with my choice of project as it was both interesting to complete whilst actually providing real benefit to people. I generally feel that the system as it stands would benefit the current paper based system used within the SoC and other external organisations and committees.

The project drew on a large range of skills gathered throughout my time at University and during my industrial placement. The modules that were of most relevance during the completion of the project were DB21, DB31 and SO22 which correspond to Database Principles and Practice, Advanced Databases and Software Project Management respectively. I believe that for a final year project to be successful it is of utmost importance to ensure that project management is given the highest priority and that the initial schedule is very carefully thought out. Whilst my schedule was by no means perfect, revisions to it were made and documented throughout the projects life cycle and this allowed adaptations to be made and control of the milestone deadlines to be maintained. It is essential for the project manager to remain in control of the project at all times.

I am pleased with the choice of methodology for the project as it allowed changes to the design to be brought about during the implementation stage and offered great flexibility when implementing the components for the system. The defined closure part of the methodology allowed for decisions to be made regarding when to release a new version of the software and hence test the full interaction of the components.

The project was approached as an experimental voyage into the often old fashioned area of meetings management and has helped pave the way to future improvements within the administration of meetings and committee procedure.

A variety of skills have been learnt or improved on during the project. These have included interview techniques, OO design, project and time management, DB design and implementation, installation and configuration of Apache Tomcat, JSP programming, Java programming and HCI issues relating to the design of a good interface.

When the project schedule was initially devised it was thought to be complete. However the project management side of the project proved demanding as other deadlines, which were not initially known, also had to be met (i.e. semester two coursework's). This meant that changes needed to be made in the project schedule and the efficiency of the plan was rather questionable in some areas.

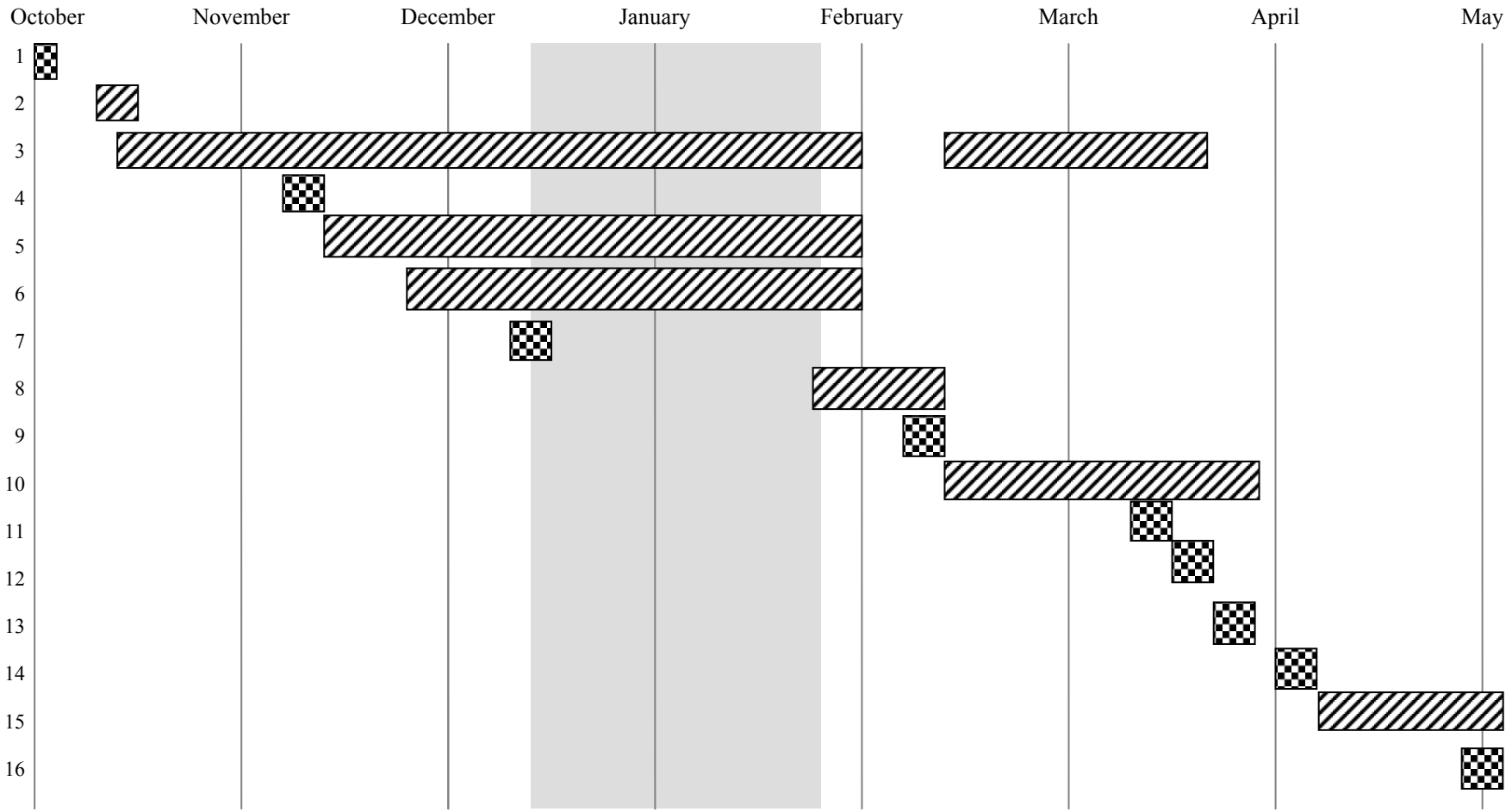
Various problems I encountered are discussed below. I have explained how I will use the experience and lessons learnt to overcome these problems in the future. This may also prove helpful for future students attempting a final year project in this subject area, or as most points are generic, it could be used by all students attempting a final year project within the SoC:

- The project plan would be devised to allow similar tasks to be completed in parallel. As separate tasks within the project can sometimes boast mutual benefit, it makes sense to conduct these at the same time.
- The project plan would allow extra time for the implementation period that was reserved for extenuating circumstances. This time could be used for issues relating to Tomcat configuration or hardware problems that inevitably delayed certain areas of my project.
- The project plan would not state exact dates for some milestones, but would instead specify a date range for the milestones to be completed during.
- I would keep a better record of progression throughout the projects lifetime and also complete versions of each chapter as and when that section of the project was finished. Having this would have inevitably allowed the final report to be constructed more efficiently.

Overall, if I were to offer any future final year students one bit of advice it would be to ensure that the projects plan/schedule is viable. It is important to try to accommodate day to day problems (such as configuration of development environment and delays with meeting third parties). I would also make sure that all revisions to the project are fully documented and any knock on effect of the delays explained.

Appendix B - Schedule

B.1 Original Schedule



Key

Milestones

Revision time inc. exams, 12th Dec – 24th Jan

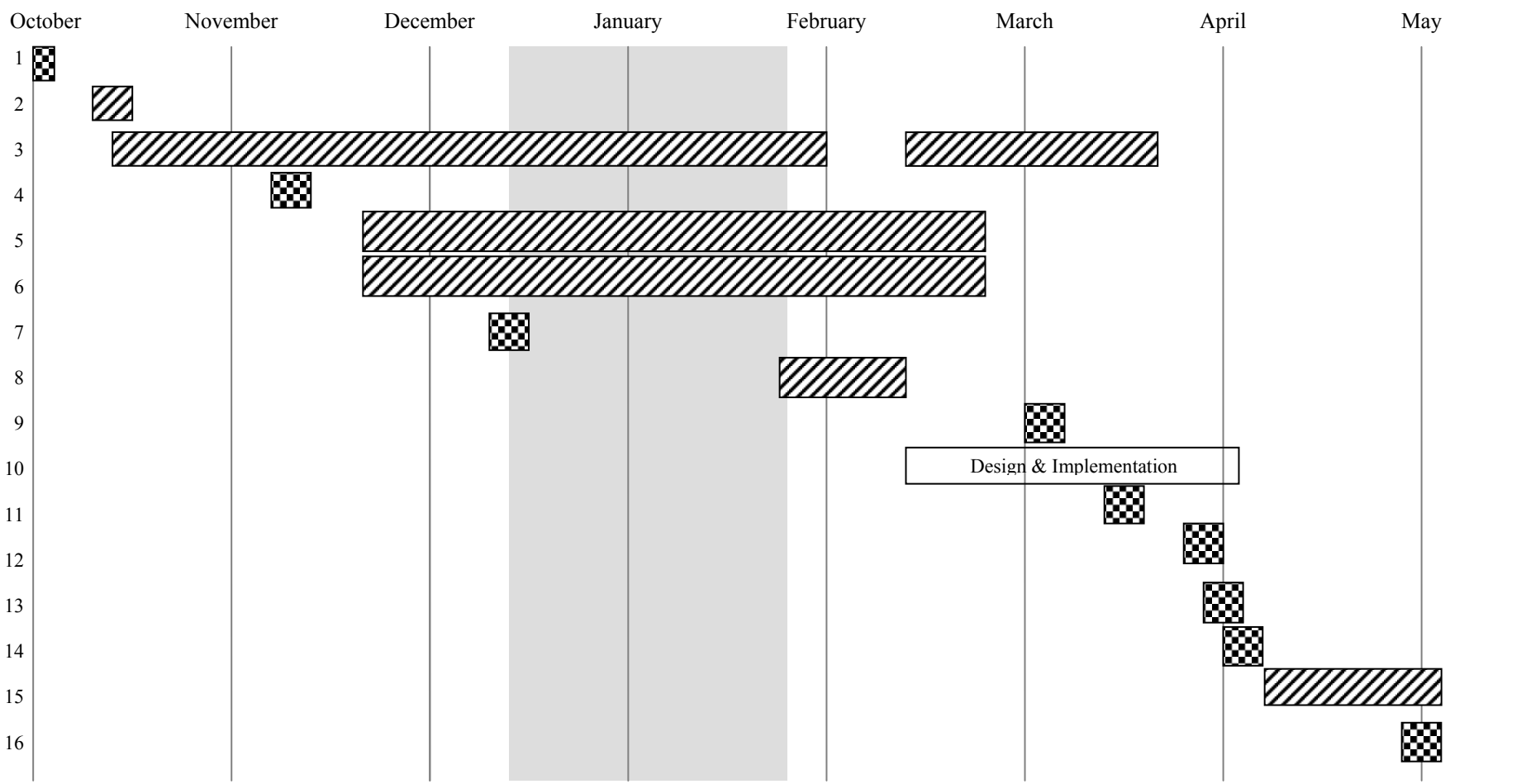
The dates below show the start of that section of the project; hence for part three, I intended to start the research on 21st October.

For Milestones, these are the exact date that the project hopes to achieve it on and are highlighted in bold below. What follows is a brief explanation of the changes made that effected these milestones or dates.

1. **October 2nd.** **Preference form submitted. This deliverable was met.**
2. October 17th Alteration of initial project idea.
3. October 21st Research.
4. **November 11th** **Aim and Minimum requirements form. This deliverable was met.**
5. November 18th Gather user requirements. Due to problems with initially setting up the first interview, this are of the project suffered delayed. It was also decided that gathering user requirements should be done in parallel with reviewing existing system as they contribute to each others success.
6. November 25th Model requirements, ensure correctness. This stage was also ongoing and involved transcripts being sent back to the interviewee to ensure they were correct.
7. **December 12th** **Mid-Project Report. This deliverable was met.**
8. January 26th Review possible methodologies. This time frame was also met.
9. **February 12th** **Complete review of existing MM systems. This deliverable was delayed due to full set of user requirements not yet being possessed. This was completed on the 6th March, a week after the last interview.**
10. February 12th Begin implementation of generic MM system. This was initially an error in the original project schedule and this time frame should have been split into design and implementation.
11. **March 14th** **Draft chapter and table of contents. This deliverable was met, although handed in a few days late by permission of supervisor.**
12. **March 21st** **Progress meeting. Unfortunately due to scheduling issues, the earliest date that could have been organised was March 31st. This essentially caused a knock on affect and caused the implementation stage to run over by a week.**
13. **March 27th** **MM system produced and evaluated – min requirements met. This was not satisfied until April 3rd. Because of the previous delay**
14. **April 5th** **Incorporate action/task management – min requirements exceeded. This deliverable was met.**
15. April 5th Complete write up of report. This stage of the project was started on time, as planned.
16. **May 2nd** **Submit final copy of report – End of project. This deliverable was met.**

B.2 Modified Schedule

A Gantt chart is shown below that helps to show the differences of the original schedule and the one which was adhered to.



Appendix C

Interview Transcripts

C.1 James Craig

Interview 01

Location: Dr Stuart Robert's Office

Time/Date: Thursday 21st November 15:15 – 15:45

Present: Josh O'Neill (Interviewer), Stuart Roberts (Supervisor), James Craig (Interviewee)

Aims of interview: The reason the interview was today rather than next week was because James Craig is away on Jury Duty. Good job I had a draft copy of the interview. Basically the main questions posed were relating to his role and how he went about doing his job.

Afterthought: Get a tape recorder for next time. Questions need slightly rewording for more technical interviewees. Prepare UML diagram to model process of document distribution and confirm it to be correct with him.

Main details of the meeting i.e. what I recorded:

James Craig (JC): He is the Senior Administrative Officer within the SoC. His role is head of Clerical and Secretarial Support.

He also chairs FEAST.

He takes minutes for the SMB (School Management Board) meeting.

This meeting takes place at least monthly, at a predetermined time. It normally has an upper bound of 2 hours. If more meetings are needed, then they are scheduled within the meeting and agreed by

majority. He takes minutes, sends out actions, if the meeting consists of presentations he includes details about these within the write up of the minutes.

The minutes are typed up using Word. He normally cuts and pastes bits out of last meetings minutes to save on the typing. They have a unique numbering used for the file names. The SMB minutes are not published to the intranet.

The circulation of the minutes is as follows:

1. Send draft version to Tony Cohn (TC), this version includes a confidential section; TC updates it and sends amendments back.
2. JC updates it and with the amendments and sends a paper copy to SMB members.
3. They suggest amendments and inform him, usually via email.
4. JC adds these amendments, removes confidential section and emails it to all staff within the school.

Even though they have had the chance to suggest possible amendments to the items within the minutes, they sometimes disagree with them at the start of the next meeting.

A way to record and monitor actions is also required.

With regards to the **agenda** the following happens:

Send draft version to Tony Cohn for first look though, he suggests some topics or items too add and sends it back to JC. JC adds the required items to the Agenda and then sends the Agenda out to all SMB members on paper. The Agenda is also then emailed to all staff.

JC keeps a local copy of the Agendas in MS word format and uses this when he prints it out, when he emails it he cuts and pastes from word to the email and sends it round.

JC believes that it would be useful for all committee members to have access to the initial draft of the agenda to see if items have been missed out.

The Apologies for absence is done by JC. JC is usually informed of people who can't attend by bumping into them in the corridor or sometimes via email. Sometimes people inform Tony Cohn instead that they are unable to attend

The thing that generally takes most time is the actual writing of the minutes.
The draft version of the agenda also takes a bit of time.

Actions from one meeting become items on the next agenda. He doesn't really use matters arising from last meeting as he believes that each action deserves to become a item in itself.

If nothing has been done relating to a specific action then this is marked as such in the minutes, i.e. it is recorded in the minutes as "Not complete".

Staff meetings – He takes minutes for the staff meetings.

These happen twice per term.
Seth Bullock is the Chair.
Very similar in structure to the SMB meetings.
The attendance is quite a bit larger as all SOC staff are invited. A post graduate representative is also present.

During the meeting a 1-2 page report called the **head of school report** is presented and is done by Tony Cohn. This is also published to the internet (although he jokingly said about 15 minutes before the meeting). Although I have it on good ground that its normally done 24 hours before hand (from the Mid Project Report).

The agenda for this differs in that it has a specific template to follow and items that people have requested to be added will only be added if they are relevant or link in to the template.

Normally the template will have the following items on:
Teaching matters
Research matters
Health and Safety – this is always on the agenda.
NB: A Use case diagram was initially developed to help with understanding the process that these admin tasks go through, the description of this use case is given below.

Use Case Description

1. Send **1st draft** to Tony Cohn (TC), TC supplies amendments and sends it back.
2. JC updates the minutes and sends it out to all SMB members (**2nd draft**)
3. The SMB members suggest amendments to JC usually via email.
4. JC updates it, removes the part for SMB members only and emails it to all SoC staff (**Minutes**).

C.2 John Davy

Interview 02 – John Davy (JD)

Location: Dr John Davy's Office

Time/Date: Tuesday 18th February 10:30 – 11:00

Present: Josh O'Neill (Interviewer), John Davy (Interviewee)

Aims of interview: To gather information about JD's role in meetings, what process/procedures he currently follows and which aspect of these could be improved to make the task of administrating meetings more efficient.

Transcript:

NB: An introduction was given before the tape recording started informing the interviewee about the aims of the interview and what my final year project was concerned with.

JO = Josh O'Neill.
JD = John Davy.

JO: What committees are you currently involved in?
JD: I am on the School Learning and Teaching Committee (SLTC) which I chair, I'm also on the school management board School management board. Staff and Student committee, Research committee which are all at school level. I am on the faculty learning and teaching committee at university level.
JO: Are you involved with the administration of meetings for these committees, if so which committees?
JD: Only the SLTC at school level.
JO: What is your role within the administration of these meetings, do you produce the minutes or the agendas etc.
JD: I produce the agendas and I check and edit the minutes, but Charlie Brown does the basic minutes.
JO: Do you have any predefined procedures that you work to at the moment with regards to producing the agenda/editing the minutes?

JD: Minutes is more straightforward, Charlie takes the minutes in the meeting and writes them up usually within one working day and emails them to me and at the first convenient opportunity, which I have to say varies with how long it takes. I check them, edit them and then send them out by email to all colleagues on the committee board who can then email back to me with regarding omissions or errors. There is a slight nuisance in that, which is something I would like to see rectified in that some of my colleagues are hardened UNIX users who do not have MS software available to them and hence cannot read MSWORD attachments and I therefore have the minor nuisance in as well as sending word attachments with the minutes on I also send a plain text copy in the body of the email, because they can't handle attachments at all. I cut and paste the text version straight into the body.

JO: Currently what software do you use for this job?

JD: I use MS word then save as plain text for the body of the email.

JO: Do you think any part of these tasks could be automated, or changed to be more streamlined. Would a central place be a good idea to hold the minutes and support documents?

JD: In principle everything should be published on the SoC website, it hasn't been recently as the person who does this has been off sick long term.

I keep my own copies of the minutes, it is always possible that I could accidentally delete or overwrite it would be good for security if there was a central repository although the website should supply that. I am thinking off the top of my head now so you have to understand that these are not necessarily the most considered views. Off the top of my head the thing that I think would be very nice, would be a structured template for minutes and agendas. I mean basically, agenda's which I do I cut and paste previous agendas to reduce the time taken to produce these agendas. To have something that provided a template would be good. It would be good if such a template didn't automatically include the date; that you could explicitly put it in because occasionally you don't spot incorrect dates when cutting and pasting. So I think there will be error reduction in that, if you were presented with a template with a field to fill in for the date and the meeting number or something like that; alternatively since the meeting numbers are in sequence if you actually had a repository that auto-generated the correct meeting number.

JO: And perhaps the date as well?

JD: Yes, and the date if you had previously entered the date for the meeting, we do actually plan the SLTC meetings at the start of the year. But occasionally we might have an extra one or cancel one, so if you did have a list of dates at the start of the year it would have to be updateable.

JO: How do you actually find out who is currently on the committee?

JD: There is an email alias that is maintained by Les Proll that contains everybody who is on the committee. I also have a word document which has a list of the committee as it is at present

and for paper distribution I give that list to the secretary that gives out/distributes the papers. There is an issue in that we have to make sure that the two coincide, that Les's alias and my paper list which just relies upon my memory about the committee changes. So there is also an issue in that we have staff and student members, staff members get paper copies of agendas and of minutes of last meeting and supporting papers that come to them via their pigeon holes. Student members don't have a pigeon hole system in quite the same way, the one in the long room is not one that I would be happy to put SLTC documents in. So we actually have them made available to them either from the office here or the front office. Arguably if you had a secure system with username and password access then they could get them all centrally.

JO: With the distribution of the supporting papers, has a case ever arisen where people have got the wrong papers or people haven't got all required papers?

JD: Yes, there was one meeting where for some reason all kinds of things went wrong regarding duplication of the papers. In general people forget papers, but I am not sure you can legislate against the fact that people forget to bring papers. One thing that is useful is that occasionally we want to refer back to a previous paper and it is not my custom to duplicate those papers again, when that happens you find that half of the committee have not got that paper with them, so what I tend to do if duplicate half a dozen spare copies. It would be quite nice if you had a central, and I am thinking of the top of my head, it would be quite nice to have a central document repository in which we put files in, following a sensible file naming convention which I will show you.

*** He handed me a template of an agenda outlining the file naming convention used both for agendas and support documents.

Each supporting document has a reference number, you can then imagine that being a link which you click on to bring the desired document up.

JO: Do you think it would be useful for clerical staff if they had an automated checklist outlining what documents and how many need to be printed out for the meeting?

JD: That would be helpful. I think something like that wouldn't do any harm at all, as it is I prepare the things for them, given them instructions about what documents need duplicating and normally they do pretty well I have to say. The time when it did go wrong was when I actually did myself, because at the last minute the secretary wasn't there., but I had also been given some rather bizarre things in the documents and I ended up doing half and the secretary did the other half and that interface didn't work. we did have a

JO: You mentioned before that you sometimes have to refer back to previous documents from old meetings, do you find that you have to refer back to previous meetings minutes?

JD: Occasionally, but it does happen. It would be quite nice to have a searchable facility to search back through previous minutes to find out what meeting we discussed a certain topic in. At the moment this can be quite cumbersome, as you can imaging "Oh yes about 18 months ago we discussed this, what was the outcome." I then have to trawl through the minutes/documents around that time based on my rough memory when it was. So some type of content addressable searching would be very nice, but that's quite sophisticated for you to provide.

JO: If members of the committee can't attend the meeting, how do they actually let you know?

JD: They email me. I usually keep the emails in a separate folder and also make a written record of who said they can't come.

JO: Is that then recorded on the minutes or the agenda?

JD: At the meeting I say I have had apologies from … Charlie then records the names and then records them on the minutes and they then show in the minutes as having had apologies. During the meeting we circulate a list of names for people to sign to show that they were there.

JO: I have been speaking to Stuart and he mentioned about the merger with the engineering faculty, do you think this will cause changes at committee level?

JD: I believe the SLTC will still deal with similar things. It is possible that the SLTC will be required to work to a standard framework and deal with the same kind of matters, perhaps have less say in what they do. I don't know, at this stage its hard to know just how much the different processes will effect they committees. The research committee will go because research will be organised differently meaning a non school based research committee.

JO: In the meetings that you are involved with, do people get given specific tasks to complete by the next meeting?

JD: There are Actions, which mainly are me but not exclusively. The actions are merely there in the meetings and not specified separately. That is a weakness, if the minutes are not checked properly people are not reminded of their actions. People are given Actions,

JO: If people do not complete the actions for the next meeting, how do they inform you?

JD: Ideally I would be emailing people to check that they have done them. On a good month I do, on a bad month I don't.

***** The Phone rings, interview terminates.

C.3 Stuart Roberts

Interview 03 – Stuart Roberts (SR)

Location: Dr Stuart Roberts Office

Time/Date: Tuesday 27th February 11:30 – 12:00

Present: Josh O'Neill (Interviewer), Stuart Roberts (Interviewee)

Aims of interview: To gather information about SR's role in meetings, what process/procedures he currently follows and which aspect of these could be improved to make the task of administrating meetings more efficient. With the final aim of acquiring user requirements/needs from the meeting.

Transcript:

NB: A brief introduction was given before the tape recording started informing the interviewee about the aims of the interview and what my final year project was concerned with although the interviewee is my project supervisor so the introduction was left very brief.

JO = Josh O'Neill.
SR = Stuart Roberts

JO: What committees are you currently involved in?
SR: I am the chair of the equal opportunities committee, I am on the SLTC chaired by John and I am also the convenor of the IS course team, which isn't really a committee I suppose.
JO: Within those committees are you involved with the administration of any of them?
SR: Yes the equal opportunities and the course team.
JO: What sort of administrative tasks do you undertake, do you produce minutes/agendas for example?
SR: For the Equal Opportunities I would normally produce the Agenda's, Judy Thursby would produce the minutes, if there were any documents I might be responsible for preparing the documents for the meetings. For the IS DB course team I produce the agendas and the minutes.
JO: What software do you currently use to produce the documents?
SR: For the IS course team I would use Word.

JO: How do you distribute the documents to the members of the committees when they are completed?

SR: Hmmm I think if I remember rightly, it's a while since I've done this. I think I would normally email them out as an attachment to those people who can read attachments and put a hard copy in the pigeon holes of those people who can't.

JO: Do they go through draft versions of the documents?

SR: They might do, yes. For the IS Course Team it might be that I would email it out and people might make amendments And then the final minutes would be produced.

JO: Does anyone else check the documents before you email them out?

SR: No.

JO: When amendments are needed do people send these to you via email?

SR: Yes, probably.

JO: Are you a member of the Software Engineering (SE) subgroup of the SLTC?

SR: Yes that's right, that's disbanded at the moment but I did chair that subgroup last summer.

JO: What ways do they interact with the parent groups?

SR: We reported back to the SLTC I think three times and that would mean producing a document that could go to SLTC to be discussed at the meeting.

JO: Do they see the minutes from the SE meetings?

SR: I made the minutes available to anybody, so I put them on the web. All the agendas and the minutes were published.

JO: With the website that you produced what was the basic aim of the website, what were your main reasons for producing it?

SR: I think probably because I am not very good with paper. My filing system tends to be not the best. So I felt that there was always a chance that I might loose the minutes or something like that. So I put it all on there, I just found it easier really to have it all electronically, or as much as I could. I couldn't have it all electronically because obviously some people produce some documents for the committee that they like in their control. But as much as possible I tried to keep it all in one place.

JO: So you found that by holding the documents centrally it helped you with your procedures and made the administration process more efficient?

SR: To have it centrally made it easier to produce the agendas and the minutes, because it meant that I could copy the agenda items across to the minutes and find related documents much quicker.

JO: Before, you said that you send out the email of the minutes to members of the committee. How do you find out who is currently on the committee?

SR: For both of those committees I would have had an alias in my email for them.

JO: How is that updated, is it updated by yourself?

SR: Yes I updated it when I realise someone has joined or left the committee.

JO: Did any other members of the committee use the website that you produced to view the minutes?

SR: To view them, I don't know. I have no idea; I didn't put a hit counter on or anything like that.

JO: With the committees that you are involved with, are the meetings pre-planned at the start of the year i.e. such things as dates, locations etc.

SR: SLTC definitely is. Equal Opportunities and the other are not. In fact for the SE, I did pre-plan those. But the other two tend to be ad-hoc.

JO: With the meetings, are there any supporting papers that need to be distributed for the meeting? If so, are these distributed to people at the meeting?

SR: For the SE I certainly did have papers that were sent out before the meeting, yeah sometimes for the Course team as well.

JO: Are the clerical staff responsible for producing these papers?

SR: I always did it myself as it's only a small committee.

JO: Do you ever need to refer back to old agendas/minutes?

SR: I did for the SE, Equal Opportunities obviously from one meeting to the next to see what actions have or haven't been done, any matters arising. For the Equal Opportunities we might occasionally need to look at past documents, although it only meets two or three times a year so its not often that a query like this will arise, but it could happen.

JO: If people can't attend a meeting how do you find out about it?

SR: Probably email,

JO: Is the apology then recorded on the agenda or the minutes, does it depend when you receive it?

SR: It would be put on the minutes.

JO: Do the people within the committees ever get given tasks or actions to do for the next meeting or a certain date?

SR: Yeah,

JO: How are they recorded?

SR: Yeah good question, for the SE committee I started to have a system where they were recorded offset in the right-hand side of the document so it was quite clear that there were actions. But because it was online I changed it to HTML which messed it up. So I can't remember now, you can have a look at them. I think they probably just appeared where they did in the minutes. I don't think I did anything special with them. I would like to have had them more obvious. I think in the past that I might have collected them together at the end of the document.

JO: At the next meeting do you then find out if people have completed the task?

SR: Yeah that would be one of the things to do,

JO: Would they sometimes email you before hand to let you know the status of the action?

SR: Well that occasionally happens, but normally they tell me in the meeting. It depends on what the action is of course. It might be something that I needed to know about well before the next meeting. Sometimes it might just be a conversation in the corridor.

JO: Do the minutes/agendas that you produce follow a set naming convention?

SR: Yeah for the SE that I did, I am not sure what Judy does for the Equal Opportunities she probably has some sort of coding system.

JO: Would the supporting papers for the meeting follow a similar naming convention?

SR: Certainly for the SE they did follow a set naming convention.

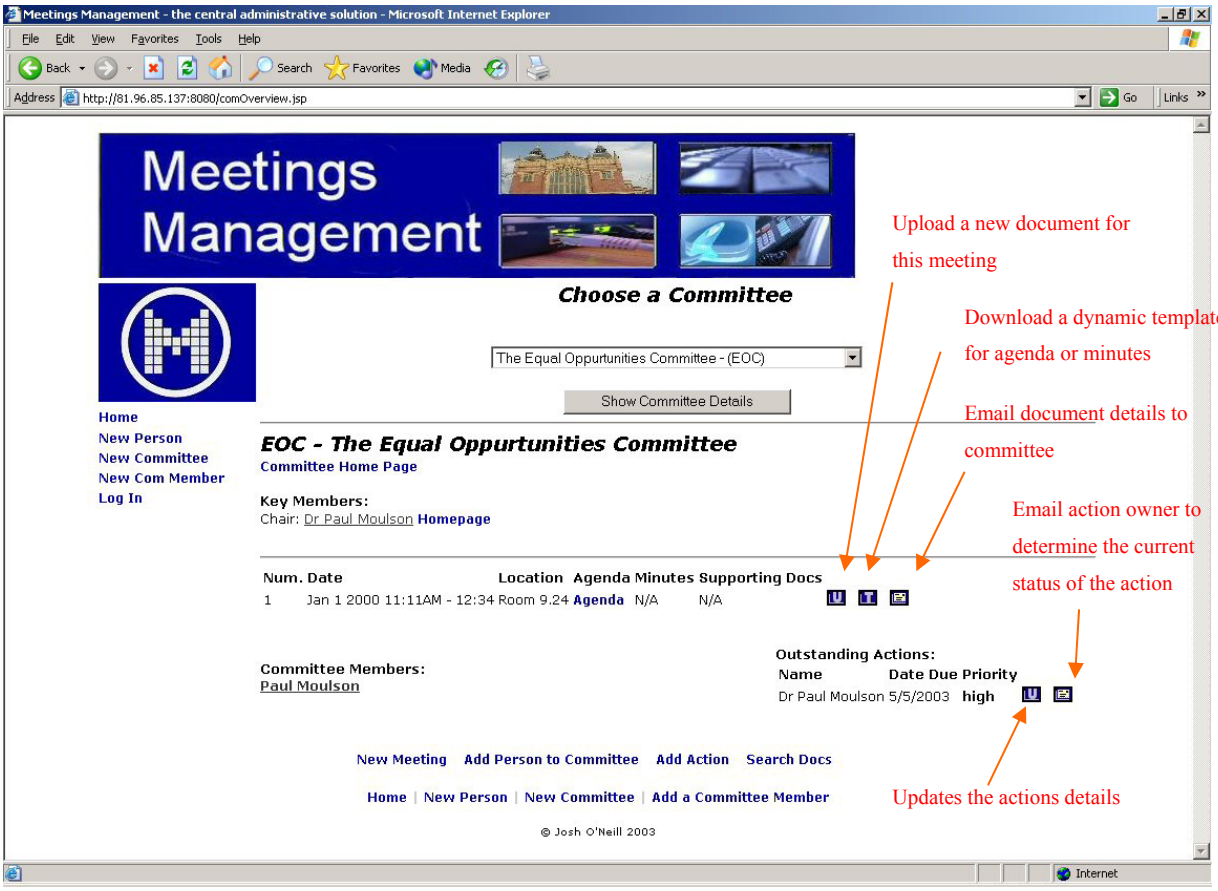
JO: Do you think it would be useful if people could view the minutes or agendas and then click on a link that would take them directly to the supporting papers?

SR: I think I would, but whether most people would I don't know. I remember when I did the SE review that somebody said that oh yes they would find It useful to look at if they lost their papers. One person said that on one occasion. I have no idea how widespread it was. You have to realise that not all supporting papers will be available 100% electronically.

Appendix D

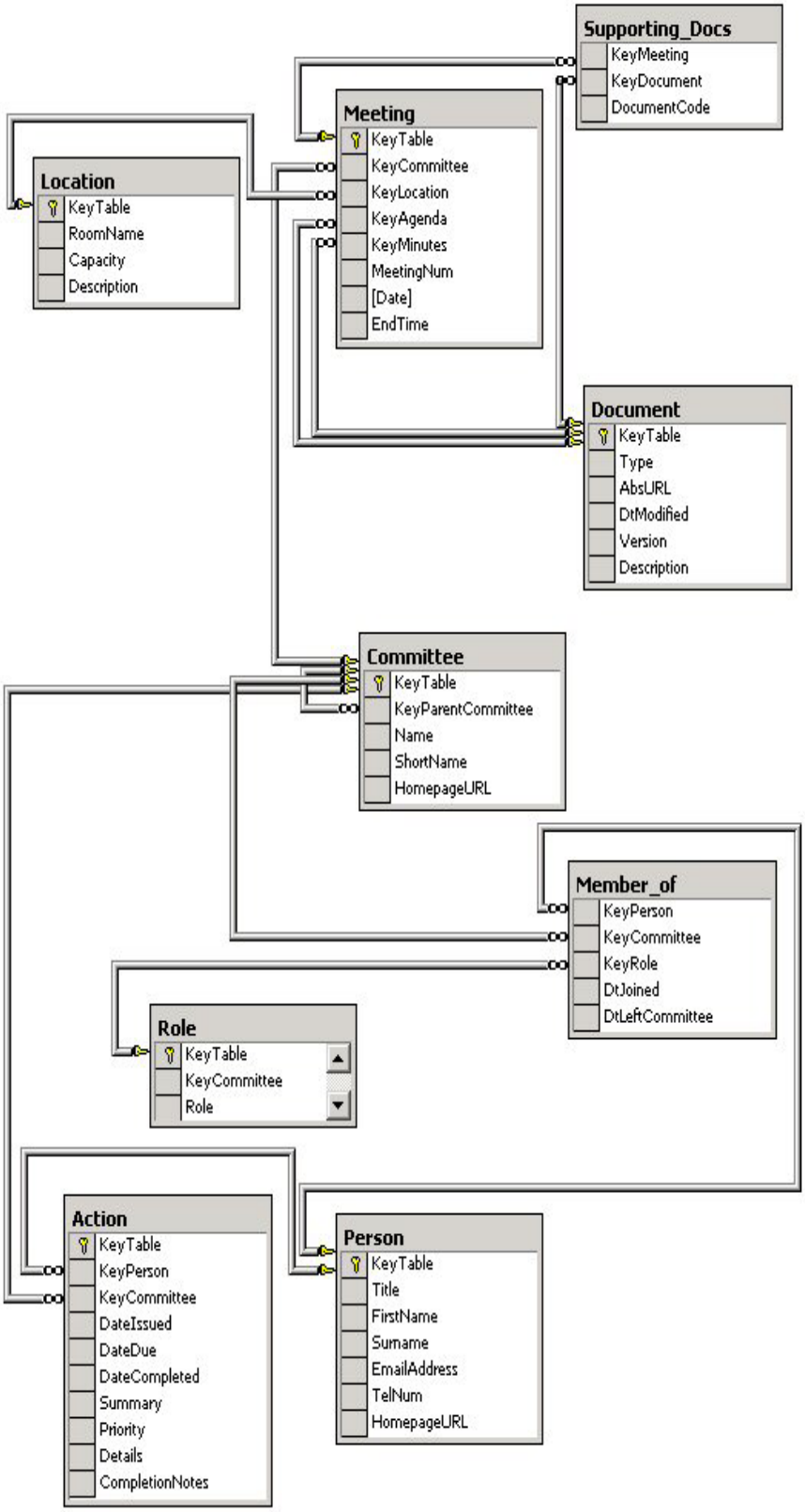
The System's Main Page

A screen shot is shown below of the main page of the system. Most functionality available within the system is via a link from this page. The less obvious functionalities have been explained in red on the screen dump.


Meetings Management - the central administrative solution - Microsoft Internet Explorer
Meetings Management
Choose a Committee
The Equal Oppurtunities Committee - (EOC)
Show Committee Details
Home
New Person
New Committee
New Com Member
Log In
EOC - The Equal Oppurtunities Committee
Committee Home Page
Key Members:
Chair: Dr Paul Moulson Homepage
Num. Date Location Agenda Minutes Supporting Docs
1 Jan 1 2000 11:11AM - 12:34 Room 9.24 Agenda N/A N/A
Committee Members:
Paul Moulson
Outstanding Actions:
Name Date Due Priority
Dr Paul Moulson 5/5/2003 high
New Meeting Add Person to Committee Add Action Search Docs
Home | New Person | New Committee | Add a Committee Member
© Josh O'Neill 2003
Upload a new document for this meeting
Download a dynamic template for agenda or minutes
Email document details to committee
Email action owner to determine the current status of the action
Updates the actions details

Appendix D

System's Main Page



Appendix F

User Manual

Minimum Requirements

Server Requirements

The computerised MM software has been designed and tested on the Apache Tomcat web server using SQL Server 2000 as the underlying DBMS. The software should run on any JSP enabled web server, but as this has not been tested it cannot be guaranteed. This manual does not discuss configuration issues with SQL Server or Tomcat and the relevant literate should be researched if problems occur with these areas. It also assumes that a JDBC-ODBC driver has been installed and that a connection can be achieved to the DB.

Client Requirements

The software should run on any PC that can run a graphical internet browser. Please note that some additional word processing software may be needed, depending on the format of the documents, people have uploaded.

The rest of this installation presumes that the CD containing the software is in your possession.

Installation

The installation of the system essentially follows three main steps, these are outlined below:

1. Creation of the DB, tables and integrity constraints
2. Populating one of the tables
3. Copying over the JSP pages, editing one class file, recompiling the JavaBeans and deploying the class files to the correct directory.

1.

Open Enterprise manager and expand the current server. Right click on the 'Databases' folder and choose New Database. Call the new DB csxjao_project.
The required DB has now been created. Tables now need to be created within this DB. This can be done by opening query analyser and running the SQL contained within the CreateTables.sql file. A user should also be set up for the database and a note should be made of the username and password.

All the required tables have now been created and it is now required that the table used for the authentication is populated with at least two rows. This leads to step 2.

2.

Query analyser should be opened, the DB called csxjao_project should be selected and the file called PopulateLogIn.sql should be opened, and then ran (or executed). Details of how to change the username and password are given in this file. The DB is now correctly configured and ready to use.

3.

The files within the 'JSPPages' directory should be copied directly to your TOMCAT_INSTALL\ROOT directory where TOMCAT_INSTALL is the absolute path for the installation of tomcat. Now the JSP pages are in the correct place, one class file needs to be edited. The folder called src should be opened and the DBConnection.java file within the src/fyproj/database directory needs to be edited. Open this file and edit the necessary lines at the top of the file. These include the serverName, databaseName (if not named as csxjao_project) and the username and password that you created in step 1. The URL and dbClass values may also require alteration, please consult your JDBC driver documentation for these details. After the necessary alterations have been made, the file needs recompiling. Recompile all the classes and copy the compiled class files into the following directory TOMCAT_INSTALL\ROOT\WEB-INF\classes this should then appear as a directory called fyproj which in turn contains directories database, report entity and util. The correct file structure is now set out on the server, all that remains is to test that it works. Simple restart the Tomcat server and open a web browser and enter the URL http://100.100.100.100:8080 where 100.100.100.100 is the IP address of the server. A webpage should then be shown asking for authentication issues. If an error occurs please contact your network administrator who may help shed some light on the problems.

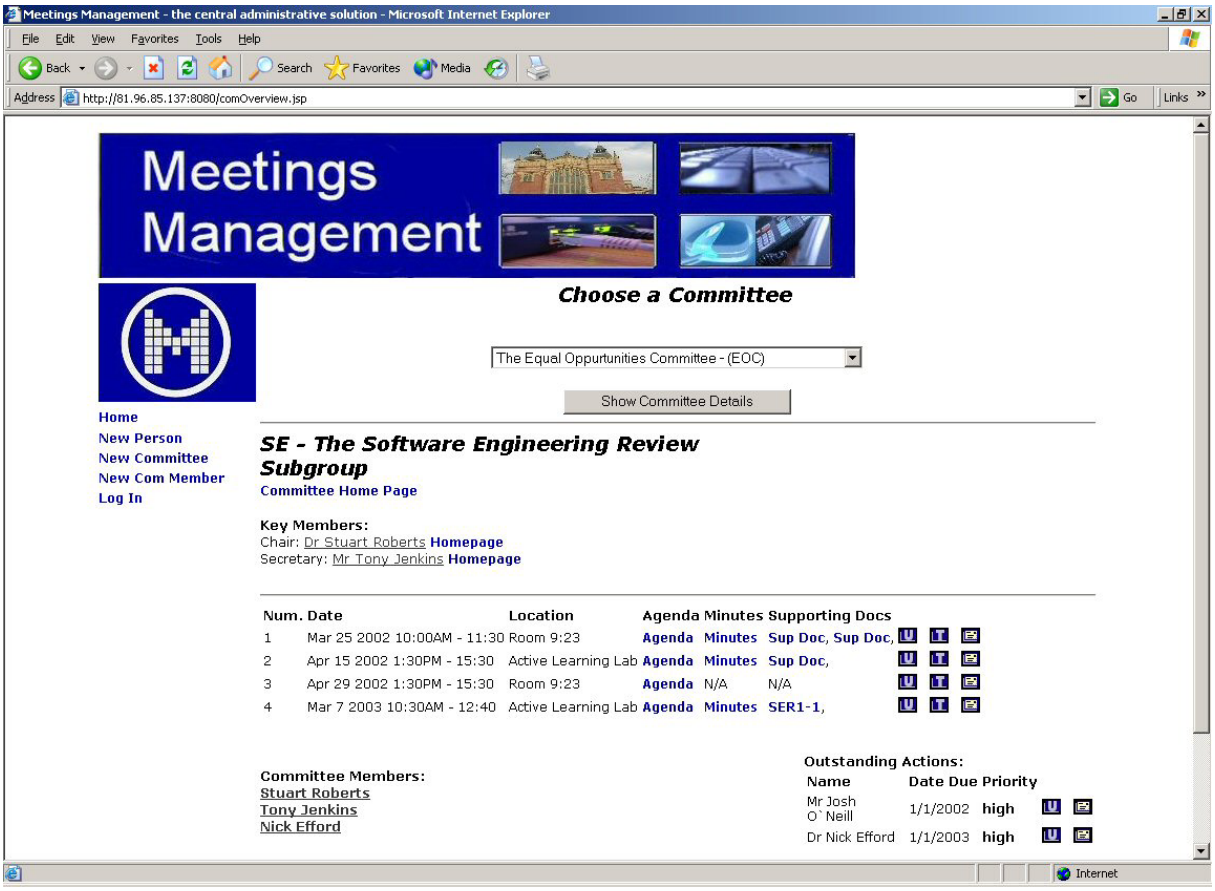
NB: If problems occur when trying to compile these files then double check that the following JAR files are available via the classpath. Servlet.jar, activation.jar and mail.jar. If they are not, these can be found in the common or lib directories of the tomcat installation. The JAR's used to allow connection to the DB should also be available via the classpath. For example if the free JDBC driver available from MS is used then, three Jars are needed. These are msutil, msbase and mssqlserver.jar respectively.

Using the System

Depending on what username or password was issued to you by the systems administrator you will either be logged on as a public user or as an admin. Initially it will not be obvious as correct

completion of a valid username and password will always take you to the main page of the system which is shown below:

Figure 1 – The main page within the MM system


Meetings Management - the central administrative solution - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://81.96.85.137:8080/comOverview.jsp
Meetings Management
Choose a Committee
The Equal Oppurtunities Committee - (EOC)
Show Committee Details
Home
New Person
New Committee
New Com Member
Log In
SE - The Software Engineering Review Subgroup
Committee Home Page
Key Members:
Chair: Dr Stuart Roberts Homepage
Secretary: Mr Tony Jenkins Homepage
Num. Date Location Agenda Minutes Supporting Docs
1 Mar 25 2002 10:00AM - 11:30 Room 9:23 Agenda Minutes Sup Doc, Sup Doc,
2 Apr 15 2002 1:30PM - 15:30 Active Learning Lab Agenda Minutes Sup Doc,
3 Apr 29 2002 1:30PM - 15:30 Room 9:23 Agenda N/A N/A
4 Mar 7 2003 10:30AM - 12:40 Active Learning Lab Agenda Minutes SER1-1,
Committee Members:
Stuart Roberts
Tony Jenkins
Nick Efford
Outstanding Actions:
Name Date Due Priority
Mr Josh O`Neill 1/1/2002 high
Dr Nick Efford 1/1/2003 high
Internet

This is the main heart of the system and most tasks stem from this page. Various functionalities are available from here. The Key Committee members are identified near the top of the page and if an email address is available for them their names are shown as underlined. Likewise if they have a homepage URL stored on the system a link is provided to allow you to visit their homepage. After this a central table is visible. This shows the details of each meeting that has been recorded on the system, as can be seen from the above screen shot, this allows locations, meeting number, date, time and location to be viewed. Hyperlinks are then provided that allow any documents that have been uploaded to the system for that particular meeting, to be viewed. When you are about to click on the hyperlink's presented to view the agenda, minutes etc. a pop-up box will appear (it appears when the mouse cursor is hovering over the link) that states the current version of the document and the summary added for that document. Towards the side of this table three icons are displayed, these correspond to the following:

- Clicking on this icon allows documents to be uploaded for that particular meeting. A new page will be displayed allowing agendas, minutes or supporting papers to be added to the system
- Clicking on this icon causes the system to dynamically create a template for that meeting's agenda or minutes. This can then be updated manually and submitted back to the server.
- Clicking this icon allows details of all the available documents for that meeting to be emailed to all committee members. A new page is displayed allowing alterations to be made to the list of recipients or to the automated contents of the message.

After this main table two smaller tables exist on this page, one shows all committee members and is situated in the bottom left hand corner of the screen shown in figure 1 and the other is on the bottom right hand side of the screen and shows the outstanding actions for the committee currently being displayed. This outlines details about each action including the action owner, the date the action is due and the priority of the action. If the mouse cursor hovers over the action row, a summary of the action is given via a pop-up box. Two icons are again reused here, these offer the following functionality:

- Clicking on this icon allows the action to be updated. This allows the action to be recorded as completed, and also allows additional notes to be added to the action to discuss any problems that might have occurred whilst trying to complete it.
- Clicking on this icon allows a reminder email to be automatically generated and sent to the action owner to ask how much progress has been made on this action. This is via another screen and it the contents or wording of the email can be changed before it is sent.

NB: It is worth noting that none of the above pages will be viewable if you are logged on in public mode. Do not worry as the system will inform you that you are logged on in public mode and will allow you to use a different username and password

In addition to this the page also offers links that allow the following to be carried out.

- **New Person**. This allows a new person to be added to the system.
- **New Committee**. Allows a new committee to be added to the system
- **New Com Member**. Allows a person within the system to become a member of a committee on the system. On the add new committee page an option is available to add a new role to the system. This role is specific to the committee and will allow for better representation of the committee structure.

- **New Meeting**. Allows a new meeting to be added to the currently viewed committee. A Date, time and location is required to add a new meeting. On the add new meeting page, a link is also available for the room booking service within the SoC. New locations can also be added from this screen.
- **Add Action**. This allows a new action to be added to a member of the currently viewed committee. This action will then show on the main page (if it has not been completed) and can be emailed to the owner or updated as necessary.
- **Search Docs**. This offers simple functionality whereby you can search for documents stored on the system by meeting number or in relation to the meetings date. This allows past documents to be retrieved.
- **Home.** No matter where you are within the system if you click the home link you will taken straight to the page shown in figure 1.
- **LogIn**. This allows you to revisit the login page and sign in as admin if you need to.

Technical Support

If you are having any troubles installing the software then please email you comments/problems to oneilldo@hotmail.com